青少年计算机学习与应用丛书



中小学教育用计算机联合设计组　编著

中华学习机“小蜜蜂-I”使用与技术参考手册

清华大学出版社

青少年计算机学习与应用丛书

中华学习机小蜜蜂-I 使用与技术参考手册

中小学教育用计算机联合设计组　编著

清华大学出版社

内 容 简 介

小蜜蜂-Ⅰ（XMF－Ⅰ）中华学习机是青少年学习计算机技术和中小学进行计算机辅助教育的有力工具。在学校中使用，配上监视器，驱动器及打印机即可组成一个较完整的计算机系统，作为学生进行计算机教育的教学设备；或配上家用电视机及盒式录音机组成基本的系统，青少年学生坐在家里就可用上计算机，进行学习、工作和娱乐。

本书分上、下两篇。上篇是中华学习机 XMF-Ⅰ 的使用说明，其中包括两部分：第一部分是 XMF-BASIC 使用说明；第二部分是 XMF-Ⅰ 汉字系统使用说明。下篇主要内容是中华学习机 XMF-Ⅰ 型微型计算机技术参考手册。

该机可供用户、青少年学生、中小学教师和计算机应用人员阅读、参考。

中华学习机小蜜蜂-Ⅰ
使用与技术参考手册

中小学教育用计算机联合设计组 编著



清华大学出版社出版
北京 清华园
清华大学印刷厂印刷
新华书店北京发行所发行



开本：787×1092 1/32 印张：10 字数：220 千字
1988 年 12 月第 1 版 1988 年 12 月第 1 次印刷
印数：00001—10000 定价：3.80 元
ISBN 7-302-00274-6/TP·104

《青少年计算机学习与应用》
丛 书 编 委 会

序

经党中央、国务院批准实施的“星火计划”，其目的是把科学技术引向农村，以振兴农村经济，促进农村经济结构的改革，意义深远。

实施“星火计划”的目标之一是，在农村知识青年中培训一批技术骨干和乡镇企业骨干，使之掌握一、二门先进的适用技术或基本的乡镇企业管理知识。为此，亟需出版《“星火计划”丛书》，以保证教学质量。

中国出版工作者协会科技出版工作委员会主动提出愿意组织全国各科技出版社共同协作出版《“星火计划”丛书》，为“星火计划”服务。据此，国家科委决定委托中国出版工作者协会科技出版工作委员会组织出版《全国“星火计划”丛书》，并要求出版物科学性、针对性强，覆盖面广，理论联系实际，文字通俗易懂。

愿《全国“星火计划”丛书》的出版能促进科技的“星火”在广大农村逐渐形成“燎原”之势。同时，我们也希望广大读者对《全国“星火计划”丛书》的不足之处乃至缺点、错误提出批评和建议，以便不断改进提高。

《全国“星火计划”丛书》编委会

1987 年 4 月 28 日

序　言

当前，世界正面临着一场新的技术革命，为了适应新技术革命的发展，我国正在大力开展普及与应用计算机技术。

目前，世界上许多国家的计算机教育的重点已从高等院校转向普通教育、职业教育，计算机正逐步形成普及的趋势。为使我国在十年或十五年以后走上工作岗位的亿万中小学生成为掌握信息社会的工具、具有计算机基础知识的科技人才，国家科委、国家教委、中国科协、电子部等单位联合组织开发了“中华学习机”，以适合中小学教学及家庭使用。这是当前世界新技术革命和教育革命的一大趋势。计算机进入学校，走向家庭，将有力地促进整个社会的进步和发展。

为了适应广大青少年学习、应用计算机的需要，我们编撰了这套“青少年计算机学习与应用”丛书。丛书以“中华学习机”系列微机为背景，除了通俗、简明地介绍计算机的使用、应用以外，还形象生动、深入浅出地介绍计算机原理、软硬件基础知识及应用发展等方面的内容。为了促进中小学生德智体美全面发展，丛书还介绍一系列辅助教育软件，其中有辅导语文、外语、数学等基础课程的学习软件，有开发青少年智力的游戏软件，还有提高文艺修养方面的艺术软件。

希望这套丛书能成为广大青少年的朋友，同时也希望它成为广大在职干部和职工的有益的参考读物。

中国计算机学会　吴几康　陈树楷

全国中学计算机教育研究中心　吕传兴　潘懋德

1987.8

目　　录

上　　篇

中华学习机 XMF-Ⅰ 型使用说明

下　篇

中华学习机“小蜜蜂-Ⅰ”（XMF-Ⅰ）型
微型计算机技术参考手册

前　言

“小蜜蜂-Ⅰ”（XMF-Ⅰ）型微型计算机属于中华学习机系列，是一种灵巧型计算机，它具有结构紧凑、体积小、软件丰富、功能强、扩充方便、一机多用、价格低廉以及性能价格比高等特点。

XMF-Ⅰ型微型计算机是专为中学、小学及幼儿园对学生进行计算机辅助教学而设计的；同时成年人也可以利用它进行自学或作为计算机教学工具；对于从事计算机应用的用户，可以利用该机的软件资源编制应用软件，或利用扩充外围接口槽开发专用的控制系统。

在 XMF-Ⅰ的设计中，采用了先进的门阵列技术，大大地提高了系统的可靠性。为适应家庭的需要，本机专门设计了适合我国 PAL 制式的彩色电视机接口和录音机接口，以便使用家用电视机作为显示器，收录机作为外存储器，从而构成一个家用的简易计算机系统；如果选用字符显示终端、软磁盘驱动器，就可以构成一个与一般计算机系统相同的计算机系统。

为了适合我国国情，可以用汉语进行辅助教学，本机设计了较强的汉字处理系统，可以在显示屏幕上显示国标一、二级汉字共 6763 个，全部用压缩点阵方式固化在 96K ROM 中，用户根据需要可显示简体或繁体字型，也可用打印机输出汉字。汉字输入方法共有六种，即：声韵法、简易拼音法、全拼音法、国标码、区位码以及形体偏旁部首法，以适合于各种不同年龄人员使用汉字的需要。

为了吸收国际、国内较流行的、适合于教学用机种的丰富软件资源，本机设计时考虑与 APPLE-II 系统相兼容。

从增强 XMF-I 型微型计算机的系统性能考虑，设计时将主存容量确定为 64K，系统硬件用只读存储器 ROM 确定为 42K，并增设了第三、四图形页面，从而使它的性能价格比高于其它同类型的机种。

"小蜜蜂-I"型微型计算机是在国家教育委员会和电子工业部领导下，由清华大学、北京师范大学、复旦大学、电子部第 48 研究所、烟台电子技术研究所组成了联合设计组，并与华南计算机公司共同协作研制。

本手册由联合设计组组织编写，参加编写的人员分工如下：使用说明部分的 XMF-BASIC 的有关章节由北京师范大学张世英编写；汉字使用有关章节由电子部第48研究所陶一峰等及烟台电子科学技术研究所郝力军、姜海宁编写；技术参考手册部分的第一、二、三章由清华大学乌振生编写；第四章由复旦大学陈光梦编写；第五章由清华大学陈在勤编写；第六章由复旦大学项长顺编写；第七章由清华大学郭仲海编写；第八章由清华大学张永路编写。

本书的总编、校由乌振生负责。

中小学教育用计算机联合设计组

上　篇

中华学习机 XMF-I 型

使用说明

第一部分　XMF-BASIC 使用说明

XMF-BASIC 是一个新开发的加强型 BASIC，固化于小蜜蜂-Ⅰ机内。它除包含 APPLEⅡBASIC 的全部命令外，又增加了一些很有用的新的命令，且对 APPLE Ⅱ BAISC的某些命令进行了扩展与修正。因此它既与APPLEⅡBASIC 兼容，又大大优于 APPLEⅡBASIC。

本手册中用了四个记号，现分述如下：

1. [ ]——这个记号中的内容可以省缺；

2. { }——这个记号中的内容，可以用一次，也可以用几次；

3. |——这个记号是“或”的意思：或用其左侧的一项，或用其右侧的一项；

4. *——由这个记号导入的叙述，为 XMF-BASIC 有别于 APPLEⅡBASIC 的部分。

由于 XMF-BASIC 仅在命令一级和常用解释入口同 APPLEⅡBASIC 的解释程序兼容，由于某些在 APPLE Ⅱ 及其兼容机上开发的应用程序（特别是调用了解释程序的某些子程序）有可能在小蜜蜂机上不能正常运行。为此，在小蜜蜂机内另置了一 APPLESOFT，它可以 100% 地运行任何在 APPLEⅡ 及其兼容机上开发的 BASIC 程

序。XMF-BASIC 中的 PCTRL-G 命令，可以很容易地将系统转为 APPLESOFT。

此外，在介绍函数的格式时，我们都用下列形式：

[行号]…函数名（自变量）

其行号用方括号括住，表明可以带有行号（用于程序执行方式），也可不带行号（用于命令执行方式）。接着用删节号，它表明这是一个函数，不是一个语句，或者说它不是一个完整的命令，而只能作为表达式中的一项。

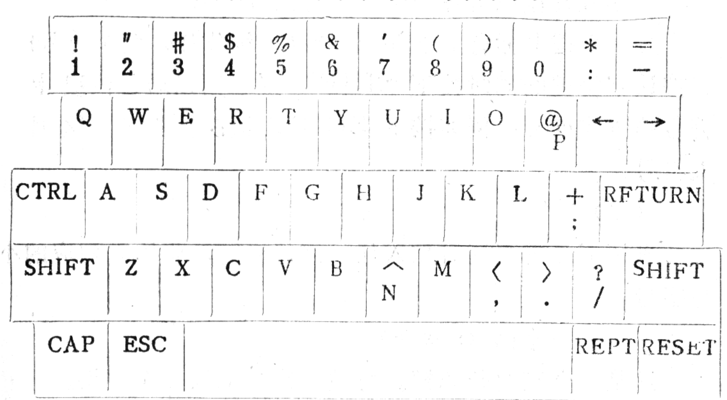
最后，还需指出，本机内置有功能较强的汉字系统，可以支持 XMF-BASIC。在进入汉字状态后，键盘、编码及内存的分配等有少许改动。有关汉字系统的说明，请参阅本篇第二部分。

第一章　键盘与显示屏

小蜜蜂计算机运行的大多数程序及其所用数据，是经由键盘敲入计算机的；而计算机接受的和输出的信息，又经显示屏显示出来。键盘与显示屏是人与机器联系的主要工具。

§ 1.1　键盘

小蜜蜂键盘共有 53 个键，如下图所示：



图中略去了输入汉字时用的特殊标记，它们将在汉字系统使用说明中详细介绍。

现就上图所示键盘分四部分作简单介绍：

一、数字键

包括阿拉伯数字的0到9。

二、英文字母键

即英文 26 个大写字母 A 到 Z。

三、特殊符号键

包括下列 23 个特殊符号：

（1）	!	（2）	"	（3）	#	（4）	$	（5）	%
（6）	&	（7）	'	（8）	(	（9）	)	（10）	*
（11）	-	（12）	=	（13）	+	（14）	^	（15）	@
（16）	;	（17）	:	（18）	<	（19）	>	（20）	/
（21）	,	（22）	.	（23）	?				

四、功能键

（一）SHIFT

此键置键盘左右各一个，具有同等效力。

凡一个键上标有上下并列（或用/线分开）两个符号时，在按该字符键的同时按下 SHIFT 键，则以敲入该键上方符号的方式处理。例如按下 SHIFT 同时按1键，这时输入符号！。

（二）CAP

当按下此键再同时敲入任一字母键时，将输入相应的小写字母。

（三）空格棒（space）

这是一个空白长条键。每按一次，输入一个空格。

（四）REPT（Repeat）

当按下此键并同时按下某一字符键时，该字符会重复作为输入，直到释放 REPT 为止。

（五）←

按此键时在显示屏上标明输入位置的光标向左移一个字符位置（一格），光标所在位置及其后面的字符将不再视作输入的内容。

（六）→

按此键时，光标右移一格，被移过的字符恢复为输入内容。

（七）RETURN

按此键时，表明一次输入内容结束，并产生一个回车换行动作。

（八）CTRL（Control）

按 CTRL 键同时按某一字母键，视为输入一个控制字符，但该字符不会被显示出来。

某些控制字符被定义产生一定的功能。比如：

1． CTRL-C 中止程序之运行。若希望继续，可用命令 CONT。

2． CTRL-G 产生“嘟”声。

3． CTRL-S 暂停显示。若希望继续显示可按任一键。

4． CTRL-X 刚刚输入的内容作废。

5． CTRL-M 相当于 RETURN 键。

6． CTRL-U 相当于→键。

7． CTRL-H 相当于←键。

（九）CTRL-RESET

将强行停止电脑的工作，但它不破坏在内存中已有的程序和数据，并返回 XMF BASIC 的等待状态。

（十）ESC（Escape）

主要用于程序的编辑。详见第十一章中 §10.16 一节。

表 1-1

键盘 ASCII 码表									
键	单独	CTRL	SHIFT	两者	键	单独	CTRL	SHIFT	两者
space	$A0	$A0	$A0	$A0	return	$8D	$8D	$8D	$8D
0	$B0	*0	$B0	—	G	$C7	$87	$C7	$87
1!	$B1	$B1	$A1	*1	H	$C8	$88	$C8	$88
2″	$B2	$B2	$A2	*2	I	$C9	$89	$C9	$89
3#	$B3	$B3	$A3	*3	J	$CA	$8A	$CA	$8A
4$	$B4	$B4	$A4	*4	K	$CB	$8B	$CB	$8B
5%	$B5	$B5	$A5	*5	L	$CC	$8C	$CC	$8C
6&	$B6	$B6	$A6	*6	M	$CD	$8D	$CD	$8D
7'	$B7	$B7	$A7	*7	N	$CE	$8E	$CE	$8E
8(	$B8	$B8	$A8	$A8	O	$CF	$8F	$CF	$8F
9)	$B9	$B9	$A9	$A9	P@	$D0	$90	$C0	$90
:*	$BA	$BA	$AA	—	Q	$D1	$91	$D1	$91
;+	$BB	$BB	$AB	*8	R	$D2	$92	$D2	*16
,<	$AC	$AC	$BC	*9	S	$D3	$93	$D3	$93
-=	$AD	$AD	$BD	*10	T	$D4	$94	$D4	*17
.>	$AE	$AE	$BE	*11	U	$D5	$95	$D5	$95
/?	$AF	*12	$BF	*13	V	$D6	$96	$D6	$96
A	$C1	$81	$C1	$81	W	$D7	$97	$D7	$97
B	$C2	$82	$C2	*14	X	$D8	$98	$D8	$98
C	$C3	$83	$C3	*15	Y	$D9	$99	$D9	*18
D	$C4	$84	$C4	$84	Z	$DA	$9A	$DA	$9A
E	$C5	$85	$C5	$85	←	$88	$88	$88	$88
F	$C6	$86	$C6	$86	→	$95	$95	$95	*19
					ESC	$9B	$9B	$9B	$9B

注：— 不产生代码

*0 产生 3 个数字 0 的编码

- •1 产生 CATALOGD1 及回车换行的编码
- •2 产生 CATALOG 的编码
- •3 产生 BRUN 的编码
- •4 产生 LOAD 的编码
- •5 产生 SAVE 的编码
- •6 产生 PR#6 的编码
- •7 产生TEXT的编码
- •8 产生 ESC D 的编码
- •9 产生 ESC B 的编码
- •10 产生 CALL-151 及回车换行的编码
- •11 产生 ESC A 的编码
- •12 产生 PRINT 的编码
- •13 产生 ESC C 的编码
- •14 产生 3D3G 及回车换行的编码
- •15 产生 3D0G 及回车换行的编码
- •16 产生 RUN 的编码
- •17 产生 LIST 及回车换行的编码
- •18 产生 BRUN 的编码
- •19 产生 8 个右移的编码

§ 1.2 键的编码和读入

用户由键盘输入时，每敲一个键，将由电路产生相应的编码。这种码称为 ASCII (American Standard Code for Information Interchange 美国信息交换标准码) 码。小蜜蜂键盘各键单独或组合成的 91 种 ASCII 码如表 1-1 所示：

在键入小写字母的状态下，大部分键的 ASCII 码与上表相同，但也有不同：

所有英文字母单独输入的 ASCII 码，需加 $20。如 a 的 ASCII 码为 E1,b 的 ASCII 码为 E2 等。此外小写字母与 CTRL 及 SHIFT 同时敲入时，某些编码为多个字符组合命令，但与本系统无关。

上表用了十六进制数。十进制的 0～9 在十六进制中仍用 0～9 表示，十进制中的 10～15 在十六进制中依次用 A，B，C，D，E，F 表示。在十六进制中，F 加 1 为 10，1F 加 1 为 20，……，FF 加 1 为 100，FFF 加 1 为 1000，……。

两位十六进制数，可表示 00～FF 范围内的数，用二进制数写出，正好是八位，即本机的一个字长，也称为一个字节。

两个字节可表示 0000～FFFF 范围内的数，它正好覆盖了本机内存的全部地址。因此，通常用两个字节来表示一个地址。

换成十进制数，一个字节可表示 0～255 的某一个数，两个字节可表示 0～65535 的某一地址。同一个地址有时也用负地址表示，只要将该地址值减去 65535 即可。

每按一个键，相应键盘 ASCII 码即送入键盘接收单元。其中最高的一位，实际上是按键时的选通信号置成的，它表明按了一个键。系统程序发现该位为 1 后，即拿去进行处理，并立即将此位（键盘接收单元最高位）置 0，以免下次误认为又按了键。这里的清 0 动作是由访问专门的清键单元完成的。下面给出键盘接收单元和清键单元的实际内存地址：

表 1-2

名　　称	内存地址		
	十进制数		十六进制数
键盘接收单元	49152	−16384	$C000
清键单元	49168	−16368	$C010

§ 1.3 屏幕方式

小蜜蜂机的显示屏具有三种方式：

一、文本方式

可显示英文字母、数字和符号，上下共 24 行，每行 40 个字符，字符由 7×5 的点阵组成。

二、低分辨率图形方式

在此方式下，显示屏幕上下分 48 行，每行 40 块，可用一定颜色将这些块涂色以成图形。

三、高分辨率图形方式

显示屏上下分成 192 行，每行分 280 个点，用一定颜色显示这些点，可构成图形。

图形和文本还可以混合。此时，显示屏下面留四行为文本方式，其余为图形方式。在混合方式下，低分辨率图形只有 40 行，而高分辨率图形只有 160 行。

§ 1.4 屏幕存储器

无论何种屏幕方式，都对应着屏幕存储器中的信息。

文本和低分辨图形方式，屏幕存储器的容量为 1024 个字节，而高分辨率图形的屏幕存储器容量则为8192个字节。

为了方便，小蜜蜂备有两个文本/低分辨率存 储 器，分别称为 1 页和 2 页；备有 4 个高分辨率存储器，分别称为 1 页、2 页、3 页、4 页（APPLE Ⅱ只有前两页）。它们的安排是如表 1-3。

可见，文本和低分辨图形的屏幕存储器是共用的。它们的第 2 页，不经特殊技巧是不能使用的。

§ 1.5 屏幕软开关

改变屏幕方式通常使用命令，也可以通过屏幕软开关来

表 1-3

屏幕方式	页号	首地址		末地址	
		十进制	十六进制	十进制	十六进制
文本/低分辨图形	1	1024	$400	2047	$7FF
	2	2048	$800	3071	$BFF
高分辨率图形	1	8192	$2000	16383	$3FFF
	2	16384	$4000	24575	$5FFF
	3	24576	$6000	32767	$7FFF
	4	32768	$8000	40959	$9FFF

改变。后者的特点是不改变屏幕存储器的信息。

屏幕软开关共有 8 个，两两一组，其内存地址及含义如表 1-4。

表 1-4

屏幕软开关		
内存地址		说明
十六进制数	十进制数	
$C050	49232 (−16304)	图形方式
$C051	49233 (−16303)	文本方式
$C052	49234 (−16302)	整幅屏幕为一种方式
$C053	49235 (−16301)	混合方式
$C054	49236 (−26300)	第一页
$C055	49237 (−16299)	第二页
$C056	49238 (−16298)	低分辨率
$C057	49239 (−16297)	高分辨率

由于小蜜蜂机增加了两个高分辨率图形页，所以又设置了一组辅助开关如表 1-5：

表 1-5

辅助屏幕软开关		
内存地址		说明
十六进数	十进数	
$C04E $C04F	49230 (−16306) 49231 (−16305)	上面表中说明不变 第 1 页改为第 3 页，第 2 页改为第 4 页

§ 1.6 文本方式

一、屏幕窗口

下面 4 个内存单元中的数据，可限定文本屏幕的大小：

32($20)：给出文本窗口左边界的列数（以 0 列为起始列）；

33($21)：给出文本窗口的宽度（字符个数）；

34($22)：给出文本窗口顶所在行数（以 1 起始）；

表 1-6

文本窗口				
功能	内存地址		最小值/正常值/最大值	
	十进数	十六进数	十进数	十六进数
左边	32	$20	0/0/39	$0/$0/$27
宽度	33	$21	1/40/40	$0/$28/$28
顶	34	$22	0/0/24	$0/$0/$18
底	35	$23	0/24/24	$0/$18/$18

35($23)：：给出文本窗口底所在行数。

这 4 个单元的最小值、正常值和最大值，见表 1-6。

二、字符显示方式

小蜜蜂机将字符送屏幕显示时，可有三种显示方式：正常的、反白的和闪烁的。这些可通过命令实现，也可通过修改字符显示方式控制单元的内容来实现。这个控制单元为内存的 50($32) 号单元。

控制码见表 1-7：

表 1-7

字符显示方式控制值

向 50($32) 号单元置数		字符显示方式
十进制	十六进制	
255	$FF	正常显示方式
63	$3F	反白显示方式
127	$7F	闪烁显示方式

三、显示 ASCII 码

送屏幕显示的任何字符（包括三种显示方式、小写字母和控制字符），都有相应的显示 ASCII 码，详见表 1-8。

四、提示符

提示符有如下 5 个，分别表明计算机工作状态，提醒用户已准备好：

]——表示浮点 BASIC（即 XMF-BASIC）状态；

>——表示整数 BASIC 状态；

*——表示监控状态；

!——表示小汇编状态；

表 1-8

显示 ASCII 码																	
		反白				闪烁				正常							
										控制字符						小写字母	
十进制		0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	十六进制	$00	$10	$20	$30	$40	$50	$60	$70	$80	$90	$A0	$B0	$C0	$D0	$E0	$F0
0	$0	@	P		0	@	P		0	@	P		0	@	P		p
1	$1	A	Q	!	1	A	Q	!	1	A	Q	!	1	A	Q	a	q
2	$2	B	R	"	2	B	R	"	2	B	R	"	2	B	R	b	r
3	$3	C	S	#	3	C	S	#	3	C	S	#	3	C	S	c	s
4	$4	D	T	$	4	D	T	$	4	D	T	$	4	D	T	d	t
5	$5	E	U	%	5	E	U	%	5	E	U	%	5	E	U	e	u
6	$6	F	V	&	6	F	V	&	6	F	V	&	6	F	V	f	v
7	$7	G	W	'	7	G	W	'	7	G	W	'	7	G	W	g	w
8	$8	H	X	(	8	H	X	(	8	H	X	(	8	H	X	h	x
9	$9	I	Y	)	9	I	Y	)	9	I	Y	)	9	I	Y	i	y
10	$A	J	Z	*	:	J	Z	*	:	J	Z	*	:	J	Z	j	z
11	$B	K	[	+	;	K	[	+	;	K	[	+	;	K	[	k	{
12	$C	L	＼	,	<	L	＼	,	<	L	＼	,	<	L	＼	l	|
13	$D	M	]	—	=	M	]	—	=	M	]	—	=	M	]	m	}
14	$E	N	^	.	>	N	^	.	>	N	^	.	>	N	^	n	~
15	$F	O	—	/	?	O	—	/	?	O	—	/	?	O	—	o	DEL

?——表示运行中要求用户输入。

此外；闪烁方块，为用户将要输入字符的显示位置。称为光标。

§ 1.7 低分辨率图形方式

文本与低分辨率图形共用屏幕存储器，前面已提及。在文本方式下，屏幕存储器中任一字节的信息总是被当作显示ASCII 码显示于屏幕的相应位置。而在低分辨率图形方式下，屏幕存储器中任一字节的信息，将不视为 ASCII 码，而是分为右 4 位和左 4 位，分别对应上下并立的两个色块的颜色编号。即按右 4 位和左 4 位的数值以相应的颜色在屏幕的相应部位涂出上下两个色块来。

由于 4 位二进制数有 16 个数值，故低分辨率图形有 16 种颜色。其对应关系见表 1-9：

表 1-9

低分辨率图形颜色编号					
编号		颜色	编号		颜色
十进数	十六进数		十进数	十六进数	
0	$0	黑	8	$8	棕
1	$1	深红	9	$9	橙
2	$2	深蓝	10	$A	灰 2
3	$3	紫	11	$B	粉红
4	$4	深绿	12	$C	淡绿
5	$5	灰 1	13	$D	黄
6	$6	中蓝	14	$E	海蓝
7	$7	淡青	15	$F	白

§ 1.8 高分辨率图形方式

高分辨率图形，颜色编号只有 8 个：

高分辨率图形颜色编号					
编号		颜色	编号		颜色
十进数	十六进数		十进数	十六进数	
0	$0	黑	4	$4	黑
1	$1	绿*	5	$5	橙*
2	$2	紫*	6	$6	蓝*
3	$3	白	7	$7	白

注：其中画*号者与显示设备有关。

高分辨率图形，是根据屏幕存储器各单元的右 7 位是否为 1 确定画点的。各单元最高位称为色选位。

实际画点时，当某位为 0，对应点为黑色（底色）；某位为 1 时，首先视其对应于屏幕的列数：处于偶列，显示紫色；处于奇列，显示绿色。但左右紧挨两点都需显示（非黑色）时，则显示白色。其次视色选位，如为 1，则紫变蓝，绿变橙。

§ 1.9 屏幕坐标系

不论绘制低分辨率图形，还是绘制高分辨率图形，都把屏幕当作一个平面直角坐标系。坐标原点（0，0）在屏幕左上角，X 轴向右，Y 轴向下（与数学中的直角坐标系上下倒向）。在低分辨图形方式下，一个色块代表 1，X 轴从 0～39，Y 轴从 0～47。在高分辨图形方式下，一个点代表 1，X 轴从 0～279，Y 轴从 0～191。

第二章　BASIC 基础

§ 2.1　BASIC 程序的构成规则

下面的例子，是一个简单的 BASIC 程序：

```
10 LET A=10
20 LET B=20.4
30 LET C=A+B
40 PRINT C
50 END
```

其中每行称作一个程序行，每行左端之数字称作行号，行号后面的词（如 LET、PRINT、END）称作关键字（或保留字），其余部分称为语句体。关键字和语句体合称为语句。

一、行号　表明电脑执行一个 BASIC 程序时的先后顺序（由小到大）。行号必须是从 0 到 63999 范围内的无符号整数。起始行号和行号间隔随意，但通常行号间隔比 1 大（上例为 10），以便在两行之间再加入程序行。

二、关键字　表明要电脑做什么事。比如 LET 表示"赋值"（如上例为让 A 取得值 10，…），PRINT 表示打印，END 表示程序结束。

关键字是系统规定的，用户不得自造。本机所用的 BASIC 关键字共 109 个，它们的功能以后会一一叙及，在此

先全部列出如下：

@	&	ABS	AND	ASC	AT
ATN	AUTO	BLOAD	BRUN	BSAVE	CALL
CHR$	CLEAR	COLOR=	CONT	COS	DATA
DEF	DEL	DIM	DRAW	EDIT	END
EXP	FLASH	FN	FOR	FRE	GET
GOSUB	GOTO	GR	HCOLOR=	HGR	HGR2
HIMEM:	HLIN	HOME	HPLOT	HTAB	IF
IN#	INPUT	INT	INVERSE	LEFT$	LEN
LET	LIST	LOAD	LOG	LOMEM:	MAT
MID$	NEW	NEXT	NORMAL	NOT	NOTRACE
ON	ONERR	OR	PCTRL-G	PDL	PEEK
PLOT	POKE	POP	POS	PRINT	PR#
READ	RECALL	REM	RESTORE	RESUME	RETURN
RIGHT$	RND	ROT=	RUN	SAVE	SCALE=
SCRN(	SGN	SHG	SHG2	SHLOAD	SIN
SPC(	SPEED=	SQR	STEP	STOP	STORE
STR$	TAB(	TAN	TEXT	THEN	TO
TRACE	USR	VAL	VLIN	VTAB	WAIT
XDRAW					

上述关键字中，@、AUTO、BLOAD、BRUN、BSAVE、EDIT、MAT、PCTRL-G、SHG 及 SHG2 等，是 APPLES Ⅱ BASIC 中所没有的。

三、语句体　表明动作的具体内容。有些语句不带语句体（如 END）。

四、BASIC 程序之规定

1．一个程序行通常含一个语句。多于一个语句时，必须在两个语句之间加分隔符冒号（:）。

2．一个语句必须在一个程序行写完，不得拆开写在两个程序行内。

XMF-BASIC 允许一个程序行可多达 239 个字符。

3．关键字必须用英文大写字母。

4．关于空格，XMF-BASIC 没有严格要求。但希望使用者不要在关键字内部加入空格，而在关键字前后各加入一个空格，这是一种好的习惯。

§ 2.2 程序的输入和执行方式

一、程序的输入

在显示 BASIC 提示符] 之后，即可输入程序。输入时，按程序行从左到右字符出现的顺序依次一一敲入，一个程序行输入完后，再敲入一次 RETURN 键（以后我们用〈CR〉代表 RETURN 键）。接着用类似的方法敲入其它程序行。

在输入一个程序行过程中，如发现敲错了，可以通过←键将光标移至出错处，重新敲入正确的内容。

输入时，不一定按行号大小的顺序，先敲入行号大的程序行再敲入行号小的程序行是可以的。

同一个行号敲了两次且其语句不同时，以后敲入的代替先敲入的。

只敲入一个行号后便按了〈CR〉键（即没有语句），程序中不会有该行程序（即使前面已经输入，也将会被删除）。

二、执行方式

XMF-BASIC 有两种执行方式：

1． 程序方式（即延迟方式）

象前面列举的那段程序，5 行全部输入后，电脑并不进行计算也不产生输出。只有敲入命令 RUN 以后，才进行计算和输出。程序执行之后，仍存留在内存中。

2． 命令方式（即立即方式）

在输入一个语句（或用冒号分隔开的多个语句）时，如果不带行号，一旦敲入〈CR〉符后，这个（或这些）语句将立即执行。执行后，这个（或这些）语句将不复存在。象上面所用的 RUN，即为立即执行方式。

§ 2.3 常量、变量及数据类型

一、常量

在程序执行过程中，永远不会改变的量，称为常量。

常量有三种数据类型：

1． 整常量（整数）

小蜜蜂允许用的整数范围为 -32767 到 +32767 之间，在这个范围之外的数，机器不能表示。产生溢出。如 234，-16898 均为合法的整数，而 66000 或 -67890 均为非法整数。

2． 实常量（实数）

实数的范围为 -10^{38} 至 $+10^{38}$，在这个范围之外的数，也为非法实数，产生溢出。当一个实数的绝对值小于2.9388×10^{-39} 时，被认为是 0。输出一个实数时，通常取 9 位有效数字。而有效位多于 9 位时，将采用科学计数法输出。科学计数法的标准格式是：

S　　×．×…×　　　E　　　S　　　T　T

符号　整数　小数部分　　　　指数符号　　指数

其中 S 表示正负号（正号可略去）；

×表示 0 到 9 的数字；

T 也为 0 到 9 的数字，表示指数，最多取两位；

E 表示 1×10 的方次，如 1000000000 可表示为 1E+9。

小蜜蜂认为下列各数均为实数 0：

.　+.　-.　.E　+.E　-.E　.E+　.E-　+.E-

+.E+　-.E+　-.E-

此外，下列各数在 INPUT 或 DATA 中也当作为实数 0：

+　-　E　+E　-E　空白　E+　E-　+E+　+E-

-E+　-E-

输出一个实数时，整数部分最左边如为正号或 0 将略去；小数部分最右边的 0 也将略去；若小数部分皆为 0 数字，小数点也予略去。

有关四舍五入，有些地方很奇妙：

PRINT 99999999.9

输出为 99999999.9

PRINT 99999999.90

输出为 100000000

PRINT 11.1111145000

输出为 11.1111115

PRINT 11.1111114519

输出为 11.1111114

如果实数的绝对值大于或等于0.01且小于999999999.2，则以定点表示法输出，即不用科学计数法。而实数的绝对值在 .0100000005 和 .0999999999 之间时，以十位数字输出（这是一个例外）。

如果你想用一个超过 38 位的数，如

211.11111111111111111111111111111111111

或

211.00000000000000000000000000000000000

均被当作溢出错误，虽然他们均在 －1E38～1E38 范围内。

*在 XMF-BASIC 中，除行号外，常数可以直接使用十六进制数，但必须由 $ 符导入。它只能表示整数，范围是 $-FFFF～$FFFF(即－65535～65535)。当一个十六进制数超过 4 位时，只取后 4 位。一个十六进制数由数字 0 开始时，左边的 0 可省去。如 $0300 可写成 $300。

3. 字符串常量

用双引号对括起来的字符序列，被称为字符串常量。字符串常量所包含的字符个数不得多于 255 个，且其中不得包含 CTRL-X,CTRL-M,CTRL-H,RETURN 等字符。

双引号对内不包含任何字符的字符串，称为空串。

二、变量

在程序运行中可以改变的量称为变量。变量实际上是常量的代号，类似于数学上用 a，b，c 代表三角形的三个边长。

XMF-BASIC 中，变量名的构成规则是：

1. 变量名的字符个数可以多至 238 个，但机器只取开始两个有效。比如 ABCD 和 ABT 均认为 AB。只用一

个字符也能构成变量名。

2. 变量名的首字符必须是英文大写字母，从第二个字符开始，可以用英文大写字母，也可以用数字，但不得用其他符号（包括英文小写字母或控制字符等）。

3. BASIC 关键字不得用作变量名或变量名的一部分。如BEND、END 均为非法变量名。

变量名同样有整型、实型和字符串型三种类型。

带有尾符 % 的变量名，称为整型变量名。如 A%，AB%，ABCD% 均是。

带有尾符 $ 的变量名，称为字符串变量名。如 A$，AB$，ABCD$ 均是。

不带尾符的变量名，称为实型变量名。

上述三种数据类型的变量，统称简单变量。

此外还有下标变量名。它的构成方法是：在上述那类简单变量名之后，加一个小括号对，括号内填入一个或多个正整型量（每两数之间用逗号隔开），这些数称作下标。如A(1，2)，它实际上类似于数学上的 a_{12}。下标的个数又称维数。一个下标变量称为一个元素。同一名字的所有下标变量又称为数组。

下标变量也有三种数据类型。其类型标志（如 $ 或 %）放在名字之后，左括号之左侧。

在本机中，如果变量名一样，但有不同的数据类型标记，可视作两个变量名。如果一个简单变量名和一个下标变量名一样，甚至类型标志也相同，仍可视为两个不同变量名。

在 XMF-BASIC 中，一个变量在未获得任何数据时，如果使用它，则取 0 值（实数型或整数型变量）或空串

（字符串型变量）。

三、数据类型的相容性和转换

整型量和实型量合称算术型量，它们之间相容。算术型量和字符串型量，是不兼容的。

一个实数转换为整数，丢弃小数部分即可。

特例：实型数 123.999999959999 转换成整型数为 123。而实型数 123.99999996 转换成整型数则为 124。

实型数 12345.999995999 转换成整型数为 12345。而实型数 12345.999996 转换成整型数为 12346。

一个整型数转换成实型数就是原来的样子。

在一个式子中既有整型数也有实型数时，总是先把所有的数当作实型数去计算。

算术型数据和字符串型数据之间的转换，详见第七章的§7.5 和 §7.6 节。

§ 2.4 函数

XMF-BASIC 共提供 28 个函数，其中 11 个为算术函数，拟在第八章专章叙述。还有 8 个与字符串有关的函数，将在第七章中介绍。其它函数则在别的章节中一一加以说明。

§ 2.5 运算符与表达式

一、运算符

1. XMF-BASIC 提供的算术运算符有

+ （加）

- （减）

* （乘）

／（除）

^ （乘方）

一个算术型量（变量、常量或具有算术值的函数）或用一个算术运算符将两个算术型量连接起来，构成一个算术表达式，它具有一个算术值。如

3 * 5

是一个算术表达式，具有算术值 15。

2. XMF-BASIC 提供一个字符串运算符

\+ （连接）

一个字符串型量（变量、常量或具有字符串值的函数）或用字符串运算符连接两个字符串型量，构成一个字符串表达式，它具有一个字符串值。如

"ABCD" + "FF"

是一个字符串表达式，其值为"ABCDFF"。

3. XMF-BASIC 提供了 6 个关系运算符

= （等于）

<> （不等于，也可写作><）

\> （大于）

\>= （大于等于，也可写作 =>）

< （小于）

<= （小于等于，也可写作 =<）

用关系运算符把两个算术表达式（或两个字符串表达式）连接起来，构成关系表达式。关系表达式具有逻辑值：或真、或假。真用 1 表示，假用 0 表示。

由于关系表达式的值为 1 或 0，通常也把关系表达式视

为算术表达式。

反过来，也可以把一个算术表达式视作一个关系表达式。在这种观点下，当算术表达式的值非 0 时，逻辑值为真，用 1 表示；当算术表达式的值为 0 时，逻辑值为假 用 0 表示。

因此

3+5>2,3

其逻辑值均用 1 表示，而

8=9,3-3

逻辑值均用 0 表示。

对两个字符串进行比较时，依次取两个字符串值的对应字符的 ASCII 码进行比较。两个字符串的值完全相同时，相等关系成立（为真），否则，在比较过程中只要发现两个字符的 ACSII 码大小不同时，其大小关系便是所属两个字符串表达式的大小关系。如

"AB"<"ABC"　　　　"AB"<"AC"

均为真，用 1 表示。

不用关系运算符连接的字符串表达式，也具有逻辑值：当该字符串表达式的值不是空串时，其逻辑值为真，用 1 表示；否则，为空串时，其逻辑值为假，用 0 表示。

4． XMF-BASIC 还提供了三个逻辑运算符

AND （逻辑与）

OR （逻辑或）

NOT （逻辑非）

把两个具有逻辑值的表达式用 AND 或 OR 连接起来，或由 NOT 后跟一个具有逻辑值的表达式，构成一个逻辑表

达式，它具有逻辑值。逻辑表达式也是算术表达式（当然更可视为关系表达式）。

逻辑运算符的运算规则是：

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

二、运算符的优先顺序

在一个表达式中，如果出现多个运算符，执行时有一定的先后顺序。而用小括号（　）括起来的部分，又先于别的部分。小括号中允许再用小括号，但这种嵌套层数不得多于36。

此外，作为一个数的正负号，也称为单目运算符。

运算符的优先顺序是：

（　）

函数

单目运算符＋、－、NOT

^

*、/

+、-

<、<=、=、>、>=、<>

AND

OR

其中排在同一行上的运算符，以出现在表达式中的顺序为序。

三、BASIC 表达式的特殊要求

BASIC 表达式中乘号必需用*，不得省去，不得用×，不得用·。除号必需用/，不能用÷，还不能将被除数和除数上下并立中间再用一横线。

对 A 的 B 次方，只能写成 A^B，或用 B 个 A连乘：A*A*A…。不能写成 A^B。

小括号必须成对出现，且不得使用方括号，或花括号。

第三章　一些和系统有关的命令

§ 3.1 LOAD 和 SAVE

命令格式：[行号] LOAD
[行号] SAVE

功能：LOAD 是从磁带上读出一个 BASIC 程序，载入电脑中。

SAVE 是将电脑中的 BASIC 程序录入磁带上。

执行 LOAD 后，电脑中原有的 BASIC 程序将被破坏，而代之以刚从磁带上装入的程序。执行 SAVE 命令后，不破坏电脑中原有的 BASIC 程序。

执行 SAVE 时，没有校对功能；执行 LOAD 时，则比较代码和，如有错将给出 ERR 信息，并中止读带动作。

执行这两个命令时，都会先后两次发出"嘟"声。

*XMF-BASIC 允许在这两个命令之后跟一个程序名，但程序名必须用双引号对括住或以双引号导入。如 LOAD"ABCD" 或 LOAD"ABCD。

带程序名的 SAVE 执行后，电脑中的程序和程序名均被录入磁带。带程序名的 LOAD 执行时，将搜索指定的程序，当读出的程序与要求的名字不符时，将提示一个闪烁的字符 N，并显示读出的程序是什么名字（若存入磁带时没有名字，则不显示名字），再继续搜索，直到读出指定的程序为

止。

不带程序名的 LOAD 和 SAVE 命令，在 XMF-BASIC 中照常使用。

用不带程序名的 LOAD，可读出在 APPLE Ⅱ 及其兼容机上存入磁带的 BASIC 程序。不论在 XMF-BASIC 下使用 SAVE 时带不带文件名，存入磁带的 BASIC 程序，可在 APPLE Ⅱ 及其兼容机上 LOAD 出来。

XMF-BASIC 在 LOAD 一个程序时，如果出现ERR 信息（多为磁带中的噪声所致），不会中止读带，而会继续搜索有效信息。

LOAD 或 SAVE 若带有程序名但又未以引号导入，则为 DOS 命令指向磁盘机，而不指向磁带。若以 LOAD 或 SAVE 作为变量名的开始部分，将会出现对磁盘或磁带的操作，这时需用 CTRL-RESET 键重新获得控制权。

§ 3.2 BLOAD、BSAVE 和 BRUN

*这三个命令是 XMF-BASIC 独有的。

命令格式：[行号] BLOAD "名字["][,A 起始地址]

[行号] BSAVE "名字["],A 起始地址，L 长度

[行号] BRUN "名字["][,A 起始地址]

功能：BLOAD 从磁带上读出指定名字的二进制文件（如机器语言程序或数据）至给定的起始地址开始的电脑内存中。如命令中略去最后一项(，A 起始地址)，则读至该文件存入磁带前在内存的区域。读出的长度取存入时的长度。

BSAVE 将电脑内存中指定的起始地址开始的、指定

长度（字节数）的二进制文件（机器语言程序或数据），以给定的名字存入磁带中。

BRUN　将磁带上指定名字的二进制文件（机器语言程序）读入由指定起始地址开始的内存中（如无最后一项，则读至存入磁带前的内存区域）并从头执行之。
三个命令执行时，均先后发出两次“嘟”声。

BSAVE 没有校对功能；BLOAD 和 BRUN 在读带时作代码和检查。读入噪音时将出现 ERR 提示，但不会中止读带，而会继续读下去。

BLOAD 和 BRUN 在读带时，将按名字搜索，找不到指定的文件时，将继续搜索下去。

起始地址和长度可以用十进制数，如 A768，或 L256，也可以用十六进制数，但必须用 $ 导入，如 A$300，或 L$100。

在三个命令中的 [″]，表明该处可用 ″ 号，也可不用 ″ 号，效果相同。而第一个″号是必须用的，否则将作为 DOS 命令，执行对磁盘的读写操作。

BLOAD、BSAVE、BRUN 均不得用作变量名的开始部分，否则也转向对磁盘的操作。

一旦错误地进入对磁盘的操作，需用 CTRL-RESET 键复位，重新获得电脑控制权。

上述五个命令 LOAD、SAVE、BLOAD、BSAVE 和 BRUN，都要对磁带机进行人工操作。在执行 SAVE 和 BSAVE 时，先接通磁带机电源，再置成录音状态（电脑对磁带机的输出口和录音口用转录线接通）；而对其它三个命令，则打开电源后，置录音机为放音状态（录音机的输

出口与电脑磁带输入口用转录线接通）。

§ 3.3 NEW

命令格式：[行号] NEW

功能：删除内存中的 BASIC 程序和所有变量。

§ 3.4 RUN

命令格式：[行号] RUN [行号]

功能：清除所有变量、堆栈，从指定的行号开始执行内存中的 BASIC 程序。如果命令中的 RUN 后不带行号，则从最小行号开始执行。

如果命令中 RUN 后跟的行号在程序中不存在或为负数，则给出出错信息。

? UNDEF'D STATEMENT EEROR

如果行号大于 63999，则会提示：

? SYNTAX EEROR

如果内存中没有 BASIC 程序，RUN 命令将立即返回到等待输入状态。一个程序经 RUN 执行结束后也返回等待输入状态。

RUN 之后带上一个名字，为 DOS 命令。

§ 3.5 STOP、END 和 CONT

命令格式：[行号] STOP

[行号] END

[行号] CONT

功能：STOP 命令会中止程序的执行，把电脑控制权

交给使用者，并提示：

BREAK IN 行号。

END 终止一个程序的执行，把控制权交还用户，无提示信息。

CTRL-C 和 CTRL-RESET 也可以中止一个程序的执行，但它们均只能用于立即执行方式，并且它们不属于关键字的范畴。

CONT 可以使 STOP、END 或 CTRL-C 中断执行的程序，从中断处的下一个命令继续执行。

如果在执行 INPUT 时由 CTRL-C 中断,则用 CONT 将会造成出错

?SYNTAX ERROR IN 行号

如果在执行 CONT 命令时机器提示

?CAN'T CONTINUE ERROR

的话，那么，在程序中止之后，你可能是

1. 改动了 BASIC 程序；

2. 做了些错误动作。

§ 3.6 TRACE 和 NOTRACE

命令格式：[行号] TRACE

[行号] NOTRACE

功能：TRACE 进入跟踪状态，显示正在执行的程序行行号。如果程序本身有输出，将和行号参杂在一起。

NOTRACE 撤消跟踪状态。

唯有 NOTRACE 或 CTRL-RESET 接着 CTRL-B 方可撤消跟踪状态。

§ 3.7 PEEK

命令格式：[行号]…PEEK（算术表达式）

功能：这是一个函数。算术表达式给出内存中一个地址，本函数的值即该地址内的数，此数用十进制表示。

*在 XMF-BASIC 中，地址若用常量给出时，可直接用十六进制数（前面冠以 $）。

§ 3.8 POKE

命令格式：[行号] POKE 算术表达式 1，算术表达式 2

功能：将算术表达式 2 的值化成二进制数存入算术表达式 1 所指的内存单元中。

算术表达式 2 的值必须在 0～255 之间。若带小数部分时，小数部分会自动舍去。

算术表达式 1 的值，必须在 －65536～65535 之间，若带小数部分会自动舍去。

*在 XMF-BASIC 中，地址（算术表达式 1）和数据（算术表达式 2）均可直接用十六进制数（前面冠以 $ 符），但十六进制数只能用正数，不能用负数。

使用 POKE 时要慎重，不要破坏系统区和程序区。企图用 POKE 向ROM 写入任何数都是无效的。

§ 3.9 WAIT

命令格式：[行号] WAIT 算术表达式 1，算术表达式 2[，算术表达式 3]

功能：允许使用者在程序中插入有条件的暂停。唯有 CTRL-RESET 可以干扰 WAIT。

算术表达式 1 是内存地址，范围为 －65535～65535。

算术表达式 2 和算术表达式 3 均为十进制数，范围在 0～255 之间。一旦执行 WAIT 时便被转换成 0～11111111 之间的二进制数。

如果 WAIT 只带两个参数（即两个算术表达式），则将算术表达式 1（地址）中的二进制数与表达式 2 所化成的二进制数逐位进行 AND(逻辑与)操作，若结果均为 0，则继续上述操作。一旦发现有 AND 结果为 1 时，执行 WAIT 后续命令（语句）。

如果 WAIT 命令带有三个参数，则先将算术表达式 1（地址）中的二进制数和算术表达式 3 所转化的二进制数逐位作 XOR 操作（称为异或操作，即对应位一致时，结果为 0，对应位不同时，结果为 1），得到一个新的二进制数，再用它和算术表达式 2 所转化的二进制数逐位进行 AND 操作。AND 结果 8 位均为 0 时，重复上面的动作。只要 AND 之后有一位非 0，则 WAIT 就算执行完结。

* WAIT 中的三个算术表达式，在 XMF-BASIC 中，均可由 $ 导入的十六进制常数表示。

§ 3.10 CALL

命令格式：[行号] CALL 算术表达式

功能：执行由算术表达式指明的内存地址开始的机器语言子程序。

这类机器语言子程序可利用系统原有的，也可以由用户

自己输入。

算术表达式的值必须在 －65535～65535 之间。

＊XMF-BASIC 允许用户用由 $ 导入的十六进制常数（$0～$FFFF 之间）来给出算术表达式。

算术表达式的值不是机器语言子程序入口地址的值时，本命令执行效果很难预测，有时不得不用 CTRL-RESET 重新获得控制权。

§ 3.11 HIMEM:

命令格式：［行号］ HIMEM：算术表达式

功能：可以由用户设定 BASIC 能使用的最高内存地址。它通常用来保护某些信息不致被破坏。算术表达式的值必须在 －65535～65535 之间，否则会出现

?ILLEGAL QUANTITY ERROR

事实上，算术表达式的取值同内存使用状态有关。通常开机时电脑会自动按内存的大小自动执行一次 HIMEM:。

HIMEM：的值被保存在内存的 116（放高位）和 115（放低位）中。执行下列命令：

```
PRINT PEEK(116)*256+PEEK(115)
```

可以打印出 HIMEM：的值来。

如果 HIMEM：的值小于 LOMEM：的值，或内存空间不够使用时，会提示

?OUT OF MEMORY ERROR

＊在小蜜蜂机上，HIMEM：的值，可由 $ 导入的十六进制数给出。

当一个 BASIC 程序在运行中，曾给某些字符串变量经

由键盘输入数据或经过连接运算符取得过数据，如果此后再执行 HIMEM：命令修改了 HIMEM：的值、则前述字符串变量将可能不会再找到。因此，通常 HIMEM：命令被用在程序头部（或先用立即执行的 HIMEM：命令）。

HIMEM:的设定不会被命令如RUN、NEW、CLEAR、DEL 或修改程序或 CTRL-RESET 等所破坏。只有重新用 HIMEM：或用 CTRL-RESET、CTRL-B 再按 RETURN 键时重新设定。但在后一种情形下原来的程序将不复存在。

§ 3.12 LOMEM:

命令格式：[行号] LOMEM：算术表达式

功能：用来设定可用的内存的最低地址，它通常是变量的首址。XMF-BASIC 自动设定在紧接 BASIC 程序之后的位址。

命令中的算术表达式的值，必须在 －65535～65535 之间。但不得比 HIMEM：的值大。

当前的 LOMEM：值，存放在内存 106 和 105 单元中，可用下面命令打印出来：

```
PRINT PEEK(106)*256+PEEK(105)
```

在一个程序内，如果两次使用LOMEM:，第二次设定的值必须大于第一次设定的值。即使这样，也会使一些变量的数据丢失，从而不能正常执行你的程序。因此，建议在一个程序中只用一次 LOMEM：命令，且写在程序的头部（或先用立即方式命令）。

LOMEM：不会被 NEW、DEL 或修改程序所破坏。

使用 CTRL-RESET 、CTRL-B 再加 RETURN 可修改 LOMEM:值，但后一种方法会删去 BASIC 程序。

*XMF-BASIC 中，LOMEM: 后面跟的算术表达式，可用由 $ 导入的十六进制数给出。

§ 3.13 USR

命令格式：[行号]…USR（算术表达式）

功能：USR 实际上是一个函数。它将算术表达式的值转入浮点累加器中，然后转内存的 $0A 执行一条转移指令（JMP），最后函数得到的值仍在浮点累加器中。

至于 $0A 开始的 JMP 应转至何处，由用户填写，通常转一个系统的入口。浮点累加器中的数，有可能在执行 JMP 时填入了新值。

这个命令要在用户对系统有较深的理解后才能用好。

§ 3.14 &

命令格式：[行号] &

功能：此命令自动转 $3F5（十进制数 1013）执行机器语言子程序。相当于执行 CALL 1013。但命令中不带参数。

在 $3F5 中通常由用户写一条 JMP 指令，转移地址由用户定。在被转移的地址，应由用户写一段机器语言子程序。

§ 3.15 CLEAR

命令格式：[行号] CLEAR

功能：清除所有变量（含下标变量）、堆栈等，但不破坏 BASIC 程序。

§ 3.16 FRE

命令格式：[行号]…FRE（表达式）

功能：这是一个函数。其自变量可以是算术表达式，也可以是字符串表达式，本身不含任何意义，是一个哑元。函数 FRE 有两个功能：

1．“大扫除”（或称“清除垃圾”）。在程序运行中，如需要经由键盘输入使某些字符串变量获得数据，或经字符串连接符的运算使某些字符串变量获得数据，这些数据将紧挨着 HIMEM：往地址较小的区域中存放。象仓库进货一样，来一批堆一批，依次堆放。这里的“仓库”称为字符串数据区。当着同一个字符串变量通过上述方法多次获得不同的数据时，这些数据都会被堆放在“仓库”里，只是位置不同而已。但这时，那个字符串变量只与最后获得的数据有关，早先存入的数据不再有效，显然这是一些无用的“垃圾”。函数 FRE 会清除这些“垃圾”，把有效数据重新整理一下，以留出多余的“仓库”空间来。

2．函数会取得的一个值，它表明自由空间还有多大（多少字节），即还有多大空间可供使用。

第四章　编辑及一些与格式有关的命令

§ 4.1　AUTO

＊这是 XMF-BASIC 增加的命令。

命令格式：AUTO　[起始行号][，行号间隔]

功能：输入一个 BASIC 程序时，用以自动生成行号。起始行号省缺时，从行号 10 开始；行号间隔省缺时，取 10 为间隔。

一行程序输入结束后，按〈CR〉键，产生下一个行号。全部程序输入结束后，用 CTRL-C 命令可退出 AUTO 状态。

在输入一个程序时，命令 CTRL-X 可以使本行程序已输入部分作废。

§ 4.2　EDIT

＊这也是 XMF-BASIC 增加的命令。

命令格式：AUTO 行号

功能：对内存中已有的程序进行行编辑。命令执行时，将指定程序行列示于屏幕顶部第二行开始，闪烁光标置于程序行之首，通过→键和←键可以移动编辑位置。

删除：将光标移至欲删除的字符位置，按 D 键即可。

连续删除若干个字符时，可连续按 D 键若干次。每删去一个字符，后续字符向左移动一个字符位置。

插入：按 I 键进入插入状态（屏幕首行提示 EDIT:I）。移动光标至插入字符位置（插入字符将在该字符之左侧），敲入被插入字符即可。改变位置再行插入时，先移动光标至插入处，敲入插入的字符即可。每插入一个字符，其后的字符均推后一个位置。退出插入状态用 ESC 键。

修改：按 C 键进入修改状态（提示 EDIT:C）。移动光标至修改字符位置，敲入新的字符即可。可连续修改多个字符，也可跳过一些字符再进行修改。退出修改状态用 ESC 键。

尾加：用X键进入尾加状态（提示 EDIT:X）。光标自动移至程序行的尾部，可在尾部增加新的内容。退出尾加用 ESC 键。

当一个程序行列出来由于填入空格而使该程序行超过 225 个字符时，EDIT 会自动改为紧密显示方式（去掉空格），可再行行编辑。

退出 EDIT，用 RETURN 键，它并不管在什么状态下，也不管光标在何位置。

欲编辑一个不存在的程序行时，只能用尾加方式由行号开始输入该程序行。

§ 4.3 DEL

命令格式：[行号] DEL 行号 1，行号 2

功能：删除从行号 1 到行号 2 之间的所有程序行。如果程序中有行号 1 或行号 2，也将被删除。

当行号 1 大于行号 2 或行号 1 至行号 2 之间（含这两个行号）没有程序行存在，命令被忽略不管。

§ 4.4 LIST

命令格式：[行号] LIST [行号 1] [，行号 2]

[行号] LIST [行号 1] [—行号 2]

功能：将程序中从行号 1 至行号 2 的所有程序行列出来。

如果省缺行号 1，则从最小行号起到行号 2 全部列出。如果省缺行号 2，则从行号 1 列至程序尾。如果两个行号都省缺，则列出全部程序。如果只有行号 1，则单独列出该行程序（没有该行程序时，命令被忽略）。LIST，等效于 LIST。

当行号 1 大于行号 2 或不存在从行号 1 到行号 2 的程序时，命令被忽略。

将程序列示出来，填入了适当的空格，因此长的程序行可能显示不出尾部一些内容（一行最多只能显示 239 个字符）。但这种显示不出的尾部，仍然存在，不会被破坏。

执行 LIST 时，文本屏幕窗口的宽度被自动设定为 33（即 $21）。

LIST 动作可被 CTRL-S 中断，再敲入任何键可继续列示。CTRL-C 和 CTRL-RESET 也可中止 LIST 的列示，但不能用 CONT 恢复列示。

* APPLE Ⅱ BASIC 不能单独列示行号为 0 的程序行（LIST 0 相当于 LIST）。XMF-BASIC 可单独将其列出。

§ 4.5 REM

命令格式：[行号] REM 字符序列

功能：这是一个非执行语句，用来为程序作一些说明，使阅读时清晰易懂。

字符序列中可以是任何字符，而以敲入 RETURN 为结束。

§ 4.6 VTAB

命令格式：[行号] VTAB 算术表达式

功能：将显示位置移到屏幕上由表达式给定的行上。

由于屏幕最上端为第一行，底部为 24 行，当算术表达式的值小于 1 或大于 24 时，会显示：

? ILLEGAL QUANTITY ERROR

* XMF-BASIC 允许在算术表达式中用 $ 导入的十六进制数。

§ 4.7 HTAB

命令格式：[行号] HTAB 算术表达式

功能：把屏幕假想成 255 个字符位置，即从 1 到 255，但屏幕行只有 40 个字符位置，因此，第 41 到第 80 个字符将实际占据下一行，第 81 到第 120 个字符占再下一行。以下类推。本命令将算术表达式的值作为位置值，使显示位置移至该处。

HTAB 0 相当于 HTAB 256

当算术表达式的值小于 0 或大于 255 时，将出错：

? ILLEGAL QUANTITY ERROR

*在 XMF-BASIC 中，算术表达式可用 $ 导入的十六进制数给出。

§ 4.8 TAB(

命令格式：[行号] PRINT TAB（算术表达式）

功能：TAB 必须用在 PRINT 语句中。同 HTAB 类似，把一行显示位置设想成从 1 到 255 个字符位置，实际上满 40 个字符时转入下一行。本命令使打印位置移至算术表达式指定的位置。但当算术表达式的值比当前显示位置还小时，则忽略 TAB 的意义。

ATB(0) 相当于 TAB(256)。

当算术表达式小于 0 或大于 255 时，为非法。

*XMF-BASIC 中，允许算术表达式用 $ 导入的十六进制数。

§ 4.9 POS (

命令格式：[行号]…POS（表达式）

功能：POS 是一个函数，作为自变量的表达式，可以是算术表达式或字符串表达式，本身无意义，只是一个哑元。本函数会给出当前显示位置的数值（为本行从左到开始的第几个字符位置）。

§ 4.10 SPC(

命令格式：[行号] PRINT SPC（算术表达式）

功能：SPC 必须用在 PRINT 语句中，用来在刚才显示位置之后加入由算术表达式给定的那么多个空格。

算术表达式的值必须在 0～255 之间。

*在 XMF-BASIC 中，算术表达式可用由 $ 导入的十六进制数给出。

§ 4.11 HOME

命令格式：[行号] HOME

功能：将文本屏幕清除，并置光标于屏幕左上角。

等效命令有 CALL-936 和 ESC@

本命令不破坏 BASIC 程序和变量。

§ 4.12 FLASH、INVERSE 和 NORMAL

命令格式：[行号] FLASH
[行号] INVERSE
[行号] NORMAL

功能：用来设定字符显示方式：FLASH 设定为闪烁方式，INVERSE 设定为反白方式。而 NORMAL 设定为正常方式。

注意：INVERSE 和 FLASH 只对输出显示有效，而 NORMAL 在输入、输出时都一样。

§ 4.13 SPEED=

命令格式：[行号] SPEED=算术表达式

功能：用来设定向外设输出字符的速度。算术表达式的值必须在 0～255 范围内。为 0 时速度最慢，为 255 时速度最快。

*在 XMF-BASIC 中，算术表达式可用由 $ 导入的十六进制数给出。

第五章　输入输出命令

§ 5.1　INPUT

命令格式：行号［“字符序列”；］变量名［{，变量名}］

功能：由键盘为所跟的变量输入数据。

INPUT 紧跟有“字符序列”部分时，这些字符将显示出来，作为输入的提示。该部分省缺时，执行 INPUT 将显示一个问号？作为提示。

INPUT 所跟的变量名有多少个，用户必须相应输入多少个数据，当输入多个数据时，可以输入一个数据敲一次〈CR〉键，如此输完；也可以输入一个数据之后敲入一个逗号（，），再如此输入第二个数，第三个数，……，最后一个数据输入后立即敲〈CR〉键。用这种“数据，数据，……，数据”序列输入的数据个数如果比 INPUT 所跟变量个数多时，将提示

？EXTRA IGNORED

但程序可以继续执行，输入的多余数据将舍弃不用。

用上述方法输入的各个数据，依顺序为 INPUT 所跟的各个变量所取得。

因此，输入各个数据时，其数据类型必须依序与 INPUT 所跟的变量名一一相容，且不得超出允许的范围。

输入的数据必须是常量，不得为表达式。

当输入的数与对应的变量名类型不相容，或输入的不是常量时，将显示

?REENTER

要你重新输入。

为字符串型变量输入数据时，可以用双引号（"）对括住，也可以不用双引号括住。

在用引号对括住的情况下，字符序列中不得再用双引号，也不允许用 CTRL-X和 CTRL-M。但可以用逗号和冒号。

如果字符序列不用双引号括住，则开始部分若有空格将被忽略，除第一个非空字符位置外，其它部位可用双引号。而逗号、冒号、CTRL-X、CTRL-M 均不得出现在字符序列中。

为一个字符串变量输入一个〈CR〉，被认为输入一个空串。

输入数据若以逗号或冒号开始，将被认为输入一个0或空串。

*XMF-BASIC 允许在执行 INPUT 时，为算术型变量用由 $ 导入的十六进制数输入非负正整数，其范围为 \$0～\$FFFF（即 0～65535）。

§ 5.2 GET

命令格式：行号 GET 变量名

功能：从键盘上为所给的变量名输入一个字符。此字符不会被显示，且输入时不需敲〈CR〉键。

当 GET 所跟变量名为字符串型时，你可以输入任何字符（字母、符号或数字），但有如下例外：

输入 CTRL-@ 会得到一个空白；

输入 CTRL-H 或← 也将得到一个空白；

输入 CTRL-C 不会中断程序的执行，而当作一个控制字符。

当 GET 后跟的变量名为算术型时，只接收输入的一个数字。当输入为

+，－，CTRL-@，E 或空格时，与输入 0 效果一样。

冒号或逗号，会输出

? EXTRA IGNORED

RETURN 键，或其它非数字键，会输出

? SYNTAX ERROR

§ 5.3 DATA 和 READ

命令格式：[行号] DATA [常量[{，常量}]

行号 READ 变量[{，变量}]

功能：执行 READ 时，所跟变量依序从 DATA 中取得数据。

DATA 中的数据，只能是常量，不能是带运算符的表达式。

READ 所带变量的数据类型必须与 DATA 中的常量的数据类型一一相容。DATA 中常量的个数不得少于 READ 中变量的个数。

DATA 可以出现在程序的任何位置。一个 DATA 语

句可以分成多个 DATA 语句来写。当 READ 所跟变量个数不止一个时，这个 READ 语句也可以写成几个 READ 语句。

DATA 中的字符串常量，可用双引号括住，也可不用双引号括住。用双引号括住时，字符串中不得再有双引号，也不能包含 CTRL-X 及 CTRL-M（即 RETURN）。不用双引号括住时，开始如有空格被忽略去，双引号可以出现在第一个非空格字符以外的任何位置，且字符序列中不得含有逗号、冒号、CTRL-X 或 CTRL-M。

* XMF-BASIC 中，允许 DATA 所带的算术常数用带 $ 的十六进制数给出，但只限于 $0～$FFFF 之间的非负数。

§ 5.4 RESTORE

命令格式：[行号] RESTORE

功能：在此命令之后，如再遇 READ，将从第一个 DATA 语句的第一个数据重新读数。

§ 5.5 LET

命令格式：[行号] [LET] 变量名 = 表达式

功能：使指定的变量按给定的表达式取得数据。

关键字 LET 可省去不写。

= 称赋值号。赋值号左部的变量与右部的表达式必须为相容的数据类型。否则将会给出错误信息：

TYPE MISMATCH ERROR

* 在 XMF-BASIC 中赋值号右部为算术表达式时，允许使用带 $ 的十六进制数。如

A = - $23 + 5:B = 5 * $10

(A 得到 - 30, B 得到 80)

§ 5.6 DEF 和 FN

命令格式：行号 DEF FN 名字（实变量名）= 算术表达式 1

行号 …FN 名字（算术表达式 2）

功能：DEF 允许用户定义一个函数名（名字，在此以 FN 导入），形式参数为实型变量名，函数的值将按算术表达式 1 的公式计算。

FN 名字，是该函数的全名称。使用时以算术表达式 2 作为实在参数，代入算术表达式 1 中计算出结果，就是该函数的值。

名字，如同实型变量名一样构成。

例子：

```
10 DEF FN A(W) = 2 * W + W
20 B = FN A(23)
30 PRINT B
```

运行后给出 69

它将由自变量 23 代替 W，按表达式 2 * W + W 算出结果为 69。

* XMF-BASIC 对自定义函数作了较大的补充：

命令格式：行号 DEF FN 名字（变量名 [{，变量名}] ）= 表达式

[行号]…FN 名字（表达式 [{，表达式}]）

功能：同上。

但这里，函数本身不限于实数型，还可以是整数型或字符串型。形式参数和实在参数的个数不限于一个，可以是多个，而且它们的数据类型可以是实型、整型或字符串型。表达式同样可以是实型、整型和字符串型。

要求形式参数和实在参数个数一致，对应数据类型相容。

例子：

```
10 X=6:Y=8
20 DEF FN A$(X,Y)=A$(X)+MID$(B$,
   Y,3)
30 A$(1)="ABCD":A$(2)="1234XY"
40 B$="HIJKLMNOPQRSTUVWXY"
50 A=2:B=3
60 PRINT FN A$(A,B+A)
70 DEF FN B(X)=X+1
80 PRINT FN B(B*B)
90 DEF FN C(X,Y,Z)=X+Y+Z
100 PRINT FN C(2.3, 3.4, 4.5)
110 DEF FN D%(X,Y)=2*X+Y
120 PRINT D%(2.2, 3.3)
130 DEF FN D(X,Y)=X*X+Y
140 PRINT FN D(Y,X)
150 END
RUN
1234XYLMN
10
```

10.2

7

70

一个自定义函数，必须先用 DEF 进行定义，然后才可使用。否则会出现错误。

?UNDEF'D FUNCTION ERROR

§ 5.7 PRINT

命令格式：[行号] PRINT [[表达式][{,|;[{表达式}]}]][,|;]

[行号] PRINT {;}

[行号] PRINT {,}

功能：打印（显示）所跟表达式的值。

关键字 PRINT 可用问号（?）代替。

PRINT 之后如果什么也不带，则产生回车换行动作（在屏幕上使光标移至下一行之首）。

PRINT 中的分号，使输出的每个数据之间不留空格连接起来。

PRINT 中的逗号，使输出的下一个数据从下一个输出区开始输出。屏幕输出区共分左中右三个区，从左端起第一个字符位置至第 16 个字符位置算第一区，从第 17 个字符位置至第 32 个字符位置算第 2 区，从第 33 个字符位置至第 40 个字符位置算第 3 区。

PRINT 之尾部若不是分号或逗号，输出结果后将自动产生换行动作。

用 PRINT 在第三区输出数据不理想，当文本窗口宽

度小于 33 时，会跑出窗口；HTAB 也会跳出窗口。有时一个数据在第三区放不下时，又会继续送至下一行第一区。

当小数点不能解释为小数点时（如 12.B），会当作 0 予以输出。

PRINT A$ + B$

如果出现提示

? STRING TOO LONG ERROR

表明两个字符串连接起来超过 255 个字符。但你用

PRINT A$ B$

则不会出现超过长度的提示。

* 在 XMF-BASIC 中，PRINT 所跟的算术表达式前面冠以两个 $ 符，则该算术表达式的值将以十六进制予以输出。

如

PRINT $$65535,$$768 + 2,$$ - 151

输出为

$FFFF　　$0300　　$FF69

这时，若该算术表达式的值有小数部分，则被舍去；若该算术表达式的值的绝对值大于 65535，仍输出十进制数。

§ 5.8 PR# 和 IN#

命令格式：［行号］ PR# 算术表达式

［行号］ IN# 算术表达式

功能：此处表达式的值为 0～7 的数，如有小数部分时则舍去。它指明槽口的编号。PR# 命令，使输出送至指定槽口，IN# 使输入由指定的槽口获得。

屏幕和键盘定义为0号槽口。

假如指定的槽口没有外设时，系统将停顿下来。除非用 CTRL-RESET 或 CTRL-C 或 RETURN 挽回。

当算术表达式的值在 8～255 之间时，会有意想不到的结果。

小蜜蜂基本型只有0号、1号5号和6号槽口，如果加上扩展箱，各槽口均可接入外设。

§ 5.9 P CTRL-G

＊这是 XMF-BASIC 独有的命令。

命令格式：P CTRL G

功能：使小蜜蜂由 XMF-BASIC 转入另一种语言系统（目前可转入 APPLE Ⅱ BASIC，以后拟转入 XMF-LOGO）。这里的 CTRL-G 是控制字符。

§ 5.10 关于声音的输出

输出一个 CTRL-G，可以产生一声"嘟"。

向 49200（$C030）置任何数，或用 PEEK 从中读数，使触发器翻转，也可发出声音，但音量很小，几乎听不到。但在短时间反复使其翻转，便可听到声音。如使翻转频率有一定变化，可奏出曲子来。

第六章 控 制 命 令

§ 6.1 GOTO

命令格式：[行号] GOTO 行号

功能：跳转到指定的行号处继续执行。

如果不存在被跳转的行号，会提示

?UNDEF'D STATEMENT ERROR IN

×××××

*XMF-BASIC 中，GOTO 所跟的行号可以用算术表达式给出。若这种算术表达式的值有小数部分，小数部分将舍去。

§ 6.2 IF…THEN

命令格式：[行号] IF 表达式 THEN 语句

功能：当表达式的逻辑值为真（用1表示）时，执行 THEN 后面的语句。否则，跳过 THEN 后面的内容，而执行下一个程序行。

当关键字 THEN 之后为 GOTO 行号时，则关键字 THEN 和 GOTO 可省缺其中任何一个。

当关键字 THEN 后为其他语句时，可以连续使用多个语句，各语句之间用语句分隔符冒号分离开即可。

在 APPLE Ⅱ BASIC 中，关键字 THEN 之左邻

若为字母 A，将会产生解释错误。如

```
10 IF B>A THEN PRINT A
```

电脑把 A 和 T 组成一个关键字，程序变成

```
10 IF B>AT HEN PRINT A
```

不但拆散了关键字 THEN，且使语句不知是何意。

*在 XMF-BASIC 中，这种解释错误是不会发生的。

§ 6.3 FOR…NEXT

命令格式：[行号] FOR 实变量名=算术表达式 1 TO 算术表达式 2 [STEP 算术表达式 3]

[行号] NEXT [实变量名[{,实变量名}]]

功能：由 FOR…NEXT 构成一个循环语句。FOR 所跟的实变量名又称为循环变量，开始执行 FOR 时，该循环变量名取算术表达式 1 的值，接着执行其后的语句，遇 NEXT 后，循环变量的当前值加上算术表达式 3 的值得到一个新值，然后视这个新值是否越过算术表达式 2 的值，若没有越过，再执行 FOR 之后的语句，遇到 NEXT 后重复上面的取新值和比较动作，直至新值越过算术表达式 2 的值之后，执行 NEXT 之后的语句。

算术表达式 1 称为初值，算术表达式 2 称为终值，算术表达式 3 称为步长或增量。STEP 算术表达式 3 缺省时，相当于 STEP 1。语句在执行时，算术表达式 1 的值，只在开始时使用一次。算术表达式 2 和算术表达式 3 均要反复使用，只要执行一次 FOR，它们的值就固定了下来，在执行 FOR…NEXT 中间是不会改变的。

在 FOR 和 NEXT 之间，可以使用各种语句，称为

循环体，循环体内又可包括 FOR…NEXT。多层 FOR…NEXT 构成循环嵌套。XMF-BASIC 允许嵌 10 次。

NEXT 后跟的实变量名，必须与对应的 FOR 所跟的实变量名一致。最内层的 NEXT 必须与最内层的 FOR 取同一实变量名……最外层的 NEXT 必须和最外层的 FOR 取同一实变量名，不得交错。不同层的 FOR…NEXT，不得共用同一实变量名。

多个 NEXT 连续出现时，可以用同一个 NEXT，跟上各个实变量名，并用逗号隔开。如：

```
100 NEXT K:NEXT J:NEXT I
```

或

```
100 NEXT K
110 NEXT J
120 NEXT I
```

可用

```
100 NEXT K,J,I
```

代替。

单一的（不嵌套的），FOR…NEXT，NEXT 后跟的实变量名可以缺省。

算术表达式 3 为正值时，算术表达式 1 不得大于算术表达式 2；算术表达式为负值时，算术表达式 1 不得小于算术表达式 2；算术表达式 3 的值不得为 0。否则循环结束条件将不能满足，造成死循环，或直至溢出出错。

如果在循环体内改变循环变量取值，要慎密考虑。当循环变量的取值向算术表达式 1 的方向倒退，且倒退长度大于或等于算术表达式 3 的绝对值时，也会造成死循环。

没有循环体的 FOR…NEXT，称为空循环，具有延时的功用。

在立即执行方式中，NEXT 要同 FOR 写在同一个输入行中，否则产生错误：

?SYNTAX ERROR

在关键字 TO 之左侧若有字母 A，输入时，T 和 O 之间不得插入空格，否则，电脑会把 A 和 T 组成关键字 AT，把句子搞乱。如

10 FOR I=A T O 1

用 LIST 列出这一行程序，变成

10 FOR I=AT O1

在 T 和 O 之间没有空格的情况下，即使 A 和 T、O和 1 之间没有空格，LIST 出来后，也是正确的。

*在 XMF-BASIC 中，三个算术表达式，均允许含有带 $ 的十六进制常数。

§6.4 GOSUB 和 RETURN

命令格式：[行号] GOSUB 行号

[行号] RETURN

功能：GOSUB 会跳转到指定行号处执行 BASIC 子程序。在执行子程序中遇到关键字 RETURN 时，将返回到调用该子程序的GOSUB 的下一个语句继续执行。

GOSUB 后跟的行号，在程序中必须存在，否则会出错：

?UNDEF'D STATEMENT ERROR IN ××××

在子程序中又可以出现 GOSUB 语句，从而构成 GOSUB 嵌套。嵌套层数不得多于 24 层。否则有出错提示

?OUT OF MEMORY ERROR

一个RETURN 找不到与之对应的 GOSUB，会有出错提示：

?RETURN WITHOUT GOSUB ERROR

注意：RETURN 是关键字，是一个个字母敲入构成的，不是 RETURN 键。

*在 XMF-BASIC 中，行号可以用算术表达式给出。这种算术表达式若有小数部分，则小数部分被舍弃。

§ 6.5 POP

命令格式：[行号] POP

功能：用在GOSUB跳入的子程序中，相当于RETURN的功能，但它不返回至调用此子程序的 GOSUB 的下一个语句，而是继续执行 POP 之后的语句。实际上，POP 相当于将调用所在子程序的 GOSUB 改变为一个 GOTO。

POP 多用于子程序嵌套中，用它撤消一个返回地址，从而使其后遇到的 RETURN 返回至较外一个 GOSUB 的下一个语句去继续执行。

在没有执行 GOSUB 语句便遇到 POP，同样会产生错误：

?RETURN WITHOUT GOSUB ERROR

§ 6.6 ON…GOTO 和 ON…GOSUB

命令格式：行号 ON 算术表达式 GOTO 行号[{，行

号}]

行号 ON 算术表达式 GOSUB 行号 [{,行号}]

功能：算术表达式的值为几，就跳转到第几个行号去。ON-GOSUB 与 ON-GOTO 之不同点是，ON-GOSUB 跳转到某一个子程序去，在那里遇到关键字 RETURN 时，将返回到 ON-GOSUB 的下一个语句继续执行；而 ON-GOTO 转走后将不再跳转回来。

ON-GOTO 和 ON-GOSUB 所跟的行号个数，不得多于 255 个。换言之，算术表达式的值必须在 0～255 之间。当算术表达式的值有小数部分时，舍去小数部分。当算术表达式的值为 0 或大于命令所跟的行号个数时，本命令被忽略，而直接执行下一个命令。

算术表达式的值小于 0 或大于 255，会提示出错：

? ILLEGAL QUANTITY ERROR

*在 XMF-BASIC 中，算术表达式中可以含有由 $ 导入的十六进制数，行号也可以由算术表达式给出。

此外，用 GOTO 命令可以代替 ON-GOTO，用 GOSUB 又可代替 ON-GOSUB。

比如

GOTO A*100+1000

可以代替

ON A GOTO 1100,1200,1300,1400,…

而

GOSUB A*100+1000

又可以代替

ON A GOSUB 1100,1200,1300,1400,…

§ 6.7 ON ERR GOTO 和 RESUME

命令格式：行号 ON ERR GOTO 行号

行号 RESUME

功能：在执行 ON ERR GOTO 之后，如果运行中出现错误，这时将不显示出错信息，也不中断停机，而是按 ON ERR GOTO 行号转至相应行号去。在那里用户可以写一段出错处理程序，处理完后加一个 RESUME 命令，它会重新返回至出错处继续运行。

例子：

```
10 ON ERR GOTO 50
20 INPUT X
30 PRINT 1/X
40 STOP
50 PRINT "DATA MUST NOT BE ZERO"
60 INPUT "X=";X
70 RESUME
```

当执行 20 行输入为 0 时，30 行不能执行，产生错误，这时不输出出错信息，也不停机，而根据第 10 行的语句，转去执行 50 行。除给出错误提示外，又由 60 行令你重新输入正确的 X 值，由 70 行控制返回 30 行输出 1/X 的值。

要求出错处理中不得再有错误。

如果在发生错误之前碰到 RESUME，则产生

? SYNTAX ERROR IN 65278

或其它怪现象。

第七章　数组与字符串函数

§ 7.1　DIM

命令格式：[行号]　DIM 变量名（算术表达式 [{，算术表达式}]）

功能：用来定义一个数组的维数和各下标的最大值。

在此，变量名又称下标变量名，也称数组名。（　）内的成分称为下标表。每个算术表达式称为一个最大下标值，算术表达式的个数称为该数组的维数。

一个数组在未经 DIM 定义时，其维数由使用语句设定，其每个下标值被自动定义为 10。

XMF-BASIC 允许维数最多为 88。最大下标变量的最大允许值依内存剩余空间大小而定。但任何最大下标值必须为正整数。如算术表达式的值含有小数部分时，丢弃小数部分。

最大下标值允许用数组元素（下标变量）给出。如果 A(5)=3，则

```
DIM B(A(5),2)
```

相当于

```
DIM B(3,2)
```

同一个数组不得用 DIM 定义两次，如果想用 DIM 定义一个已使用过的数组，也被当作第二次定义。这时会产生

出错信息：

? REDIM'D ARRAY ERROR

数组的下标值，从 0 开始。未定义的一维数组有 11 个元素，未定义的二维数组共 11 * 11 个 121 个元素。而用 DIM 定义过的数组，其元素个数为

（算术表达式 1+1）*（算术表达式 2+1）*……*（算术表达式 n+1）

其中算术表达式 i 为第 i 个最大下标值。

§ 7.2 MAT（或 @）

*这是 XMF-BASIC 增加的命令。关键字 MAT 可用 @ 代替。

@ 命令不能单独使用，只能由它导入一个赋值语句或打印语句。

用于赋值语句中，可以实现数组间的整体赋值和 +、- 运算（字符串型数组只能作 + 运算）。用于 PRINT 中，则可输出数组的所有元素。

例子：

```
    10 FOR I=0 TO 10:A(I)=I:NEXT
    20 @ B=A:@D=C-B
    30 @PRINT D;
    40 END
  RUN
  0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10
```

执行第 10 行时，数组 A 的各元素被赋值（依次为 0，1,2,…,10），执行 20 行时，先将数组 A 各元素赋值给数

组 B，再使数组 C 各元素与数组 B 各元素相减，之后赋值给数组 D。由于数组 C 未赋过值，均取 0，于是D中各元素应是数组 B 各元素的负数。经 30 行整体打印后，输出上列结果。

赋值号左部与右部的数组必须具有相同的维数及相同的最大下标值。这类语句中的任一数组如未定义过也未使用过，其维数和各最大下标值依本语句中已定义过的或已使用过的数组取得一致。如果这类赋值语句中的所有数组均未被定义或使用过，则隐含为二维数组，两个最大下标值均为 10。

在 @ PRINT 中，数组必须定义过或使用过。

§ 7.3 STORE 和 RECALL

命令格式：[行号] STORE 算术型数组名

[行号] RECALL 算术型数组名

功能：STORE 可以把指定的数组录于磁带上，而 RECALL 可以把磁带上存录的数组元素读至电脑中指定名字的数组。

读出时的数组名，可以不用存入时的数组名。但 RECALL 所跟数组名的维数及最大下标值最好与 STORE 所跟数组一致，否则会使读出的数组中有杂乱数字和许多 0，或发生

? OUT OF MEMORY ERROR

当存入的元素个数少于 RECALL 所跟数组的元素个数时，会使读出的元素中有杂乱的数字和许多 0。

反之，当存入的元素多于 RECALL 所跟数组的元素个数时，又会造成

? OUT OF MEMORY ERROR

当存入的数组的维数与 RECALL 所跟数组维数不同时，会产生信息

ERR

在读写数组时，都会先后发出两次“嘟”声。

STORE 和 RECALL 只能被 CTRL-RESET 打断。

§ 7.4 LEN

命令格式：[行号]…LEN（字符串表达式）

功能：这是一个函数，给出字符串表达式的值的长度（字符个数）。

如果字符串表达式的值超过 255 个字符，则有出错提示：

? STRING TOO LONG ERROR

§ 7.5 STR$

命令格式：[行号]…STR$（算术表达式）

功能：这是一个函数，它可以把算术表达式的值变成字符串。

STR$(3+2.5) 的值是字符串 5.5。

STR$(100000000000) 的值是字符串 1E+11。

当算术表达式的值超过实数范围时，将有出错提示？

? OVERFLOW ERROR

* 在 XMF-BASIC 中，允许算术表达式中有 $ 导入的十六进制数。如：

```
PRINT STR$(3*(-$20))
```

输出为 -96。

§ 7.6 VAL

命令格式：[行号]…VAL（字符串表达式）

功能：这是一个函数，用来将字符串表达式的值转化成算术常数。

如果字符串表达式的值第一个非空格字符不是数字的话，函数取 0 值。否则逐一检查，直至遇到不能解释成数的字符（空格、小数点、+、-、E 可解释成数）为止，将检查过的内容解释成数。

当字符串表达式的值超过 255 个字符时，提示出错信息

? STRING TOO LONG ERROR

当被解释成的数超过 1E38 或超过 38 位时，会产生出错信息

? OVERFLOW ERROR

§ 7.7 CHR$

命令格式：[行号]…CHR$（算术表达式）

功能：这是一个函数，它可以把算术表达式的值当作 ASCII 码，取得该码的相应字符。如：CHR$(65) 的值为字符 A。

算术表达式的值，必须在 0～255 之间。否则会提示出错

? ILLEGAL QUANTITY ERROR

如果算术表达式的值有小数部分，则丢弃小数部分。

* 在 XMF-BASIC 中，算术表达式可用十六进制数表示，范围为 $0～$FF。

§ 7.8 ASC

命令格式：[行号]…ASC（字符串表达式）

功能：这是一个函数，它的值为指定字符串表达式的值中第一个字符的机内 ASCII 码的十进制数。如：ASC（"ABC"）的值为 65。

如果把 CTRL-@ 用于 ASC 函数中，会有错

？SYNTAX ERROR

§ 7.9 LEFT$

命令格式：[行号]…LEFT$（字符串表达式，算术表达式）

功能：这是一个函数，字符串表达式的值称作源串，假定算术表达式的值为 M，那么本函数的值为源串中开始的M个字符所组成的子字符串。比如

PRINT LEFT$("1234ABC", 4)

输出为 1234。

算术表达式的值必须在 1～255 之间，若有小数部分时，丢弃小数部分。

算术表达式的值比源串长度大时，函数取源串的全部字符。

*在 XMF-BASIC 中，算术表达式可用十六进制数给出，其范围为 $1～$FF。

§ 7.10 RIGHT$

命令格式：[行号]…RIGHT$（字符串表达式，算术表达式）

功能：这是一个字符串函数。其中字符串表达式的值叫

源串。设算术表达式的值为 M，那么，本函数的值为从源串中取右边的 M 个字符所组成。如

PRINT RIGHT$ ("123ABC, 4")

输出为 3ABC。

算术表达式的值必须在 1～255 之间，若有小数部分时予以舍弃。

算术表达式的值比源串长度大时，函数取源串的全部字符。

*XMF-BASIC 中，算术表达式可用十六进制数给出，其范围为 $1～$FF。

§ 7.11 MID$

命令格式：[行号]…MID$（字符串表达式，算术表达式 1，[算术表达式 2]）

功能：这是一个字符串函数。自变量中字符串表达式称作源串。设算术表达式 1 和算术表达式 2 的值分别为 M 和 N，则本函数为从源串中第 M 个字符起向右取 N 个字符所组成的子字符串。

当 N（算术表达式 2）被省缺时，函数取从源串中第 M 个字符开始到尾部的字符所组成的子字符串。

当 M+N 超过源串长度或大于 255 时，多余的位置被忽视。

但 M 和 N均须在 1～255 之间，如有小数部分时，则丢弃小数部分。

*XMF-BASIC 中，算术表达式 1 和算术表达式 2 均允许用十六进制数给出。范围为 $1～$FF。

第八章　算术函数

§8.1　固有函数

一、SIN

格式：[行号] … SIN（算术表达式）

功能：求算术表达式值的正弦值。其中自变量（即算术表达式）是弧度值。

二、COS

格式：[行号] … COS（算术表达式）

功能：求算术表达式值的余弦值。其中自变量（即算术表达式）是弧度值。

三、TAN

格式：[行号] … TAN（算术表达式）

功能：求算术表达式值的正切值。其中自变量（即算术表达式）是弧度值。

四、ATN

格式：[行号] … ATN（算术表达式）

功能：求算术表达式值的反正切值，以弧度表示。

五、INT

功能：求不大于算术表达式的最大整数。

在 APPLE Ⅱ BASIC 中，INT 函数有时结果有错。比如

```
PRINT INT(3.3*100)
```

输出为 329。正确的结果应为 330。

*XMF-BASIC 改正了类似的错误。

六、SGN

格式：[行号] … SGN（算术表达式）

功能：求算术表达式的符号：

当算术表达式的值>0 时，函数值为 1；

当算术表达式的值=0 时，函数值为 0；

当算术表达式的值<0 时，函数值为 -1；

七、ABS

格式：[行号] … ABS（算术表达式）

功能：求算术表达式的绝对值。

八、SQR

格式：[行号] … SQR（算术表达式）

功能：求算术表达式的平方根。算术表达式的值不得为负。

九、EXP

格式：[行号] …EXP（算术表达式）

功能：这是一个指数函数，求以 e 为底，指数为算术表达式的值（其中，e=2.178289）。

十、LOG

格式：[行号] … LOG（算术表达式）

功能：求算术表达式的自然对数。

十一、RND

格式：[行号] … RND（算术表达式）

功能：函数取从 0 到 1 之间的伪随机数。

但可为 0，而不可为 1。

当算术表达式的值大于 0 时，则每次使用 RND 函数，取得一个新的随机数。

当算术表达式的值小于 0 时，则每次使用 RND 函数，总是依算术表达式取值，从某一个特定随机数开始产生随机数。

当算术表达式的值为 0 时，使用 RND 将得到刚刚产生过的随机数。

*在 XMF-BASIC 中，上述各函数的自变量（即算术表达式），均可含由 $ 导入的十六进制数。

§ 8.2 衍生函数

下面的函数，在 XMF-BASIC 中（包括 APPLE Ⅱ BASIC 中）并不存在，但是可以利用已有的函数，用 DEF FN 来定义。

1. SECANT（正割）

   SEC(X)=1/COS(X)

2. COSECANT（余割）

   CSC(X)=1/SIN(X)

3. COTANGENT（余切）

   COT(X)=1/TAN(X)

4. INVERSE SINE（反正弦）

   ARCSIN(X)=ATN(X/SQR(-X*X+1))

5. INVERSE COSINE（反余弦）

   ARCCOS(X)=-ATN(X/SQR(-X*X+1))+1.5708

6. INVERSE SECANT (反正割)

ARCSEC(X) = ATN(SQR(X * X - 1)) +
(SGN(X) - 1) * 1.5708

7. INVERSE COSECANT (反余割)

ARCCSC(X) = ATN (1/SQR (X * X - 1)) +
(SGN(X) - 1) * 1.5708

8. INVERSE COTANGENT (反余切)

ARCCOT(X) = - ATN(X) + 1.5708

9. HYPERBOLIC SINE (双曲线正弦)

SINH(X) = (EXP(X) - EXP(- X))/2

10. HYPERBOLIC COSINE (双曲线余弦)

COSH(X) = (EXP(X) + EXP(- X))/2

11. HYPERBOLIC TANGENT (双曲线正切)

TANH(X) = - EXP(- X)/(EXP(X) +
EXP(- X)) * 2 + 1

12. HYPERBOLIC SECANT (双曲线正割)

SECH(X) = 2/(EXP(X) + EXP(- X))

13. HYPERBOLIC COSECANT (双曲线正割)

CSCH(X) = 2/(EXP(X) - EXP(- X))

14. HYPERBOLIC COTANGENT

(双曲线余切)

COTH(X) = EXP (- X) / EXP (X) -
EXP(- X) * 2 + 1

15. INVERSE HYPERBOLIC SINE

(反双曲线正弦)

ARGSINH(X) = LOG(X + SQR(X * X + 1))

16. INVERSE HYPERBOLIC COSINE

(反双曲线余弦)

ARGCOSH(X)=LOG(X+SQR(X*X-1))

17. INVERSE HYPERBOLIC TANGENT

(反双曲线正切)

ARGTANH(X)=LOG((1+X)/(1-X))/2

18. INVERSE HYPERBOLIC SECANT

(反双曲线正割)

ARGSECH(X)=LOG((SQR(-X*X+1)+1)/X

19. INVERSE HYPERBOLIC COSECANT

(反双曲线余割)

ARGCSCH(X)=LOG(SGN(X)*SQR(X*X+1)+1)/X

20. INVERSE HYPERBOLIC COTANGENT

(反双曲线余切)

ARGCOTH(X)=LOG((1+X)/(X-1))/2

21. A MOD B (模数)

MOD(A)=INT(A/B-INT(A/B))*B+0.05)*SGN(A/B)

第九章　绘图与游戏控制命令

§ 9.1　GR

命令格式：[行号]　GR

功能：进入低分辨率图形与文本混合方式。图形占 40 ×40 个色块位置，并清除图形画面（涂成黑色），屏幕底部留 4 行文本窗口，光标置文本窗口内。

执行 GR 之后，自动设置 COLOR＝0。

当某一变量名以 GR 开始时，先执行 GR，再执行其余部分，会导致错误

?SYNTAX ERROR

§ 9.2　COLOR＝

命令格式：[行号]　COLOR＝ 算术表达式

功能：设定低分辨图形的颜色编号（参阅第一章 §1.7）

算术表达式的值必须在 0～225 之间，但超过 16 时减去 16，直至取 0～15 之间的数为颜色编号。

*XMF-BASIC 中，算术表达式可以用十六进制数，其范围为 $0～$FF。

§ 9.3　PLOT

命令格式：[行号]　PLOT 算术表达式 1，算术表达

式 2

功能：在横坐标为算术表达式 1 的值、纵坐标为算术表达式 2 的值的位置上用刚刚由 COLOR= 设定的颜色，画出一个低分辨率的色块来。

这里算术表达式 1 的值在 0～39 之间，算术表达式 2 的值在 0～47 之间，否则会显示出错

? ILLEGAL QUANTITY ERROR

在混合方式下，算术表达式 2 的值最好只用 0～39 之间的数，当用 40～47 之间的数时，会以字符的方式显示于底部的文本行中。

本命令在高分辨图形方式中无效。

* 在 XMF-BASIC 中，算术表达式均可用十六进制数给出。

§ 9.4 HLIN

命令格式：[行号] HLIN 算术表达式 1，算术表达式 2，AT 算术表达式 3

功能：设三个算术表达式的值分别为 N1，N2 和 N3，则命令会以由 COLOR= 设定的颜色，在坐标点（N1,N3）和坐标点（N2,N3）之间画一条低分辨率水平色带。

这里的 N1 和 N2 必须在 0～39 之间。N3 必须在 0～47 之间，但 N3 为 40～47 之间的数时，将会在文本行中以字符形式予以显示。

本命令在高分辨图形方式中无效。

* 在 XMF-BASIC 中，三个算术表达式均可用带 $ 的十六进制数给出。

§ 9.5 VLIN

命令格式：[行号] VLIN 算术表达式 1，算术表达式 2，AT 算术表达式 3

功能：设三个算术表达式的值分别为 N_1，N_2 和 N_3，那么，命令执行后，会以由 COLOR＝ 刚设定过的颜色，在坐标点（N_3，N_1）和坐标点（N_3，N_2）之间画一条低分辨垂直色带。

这里的 N_3 必须在 0～39 之间，N_1 和 N_2 必须在 0～47 之间，否则会有出错提示

?ILLEGAL QUANTITY ERROR

而 N1 或 N2 在 40～47 之间时，文本行会出现一些字符。

本命令在高分辨图形方式中无效。

*XMF-BASIC 允许三个算术表达式用带 $ 的十六进制数。

§ 9.6 SCRN (

命令格式：[行号] … CSRN（算术表达式 1，算术表达式 2）

功能：这是一个函数，它的值为 0～15 的一个整数，取坐标为（算术表达式 1，算术表达式 2）的颜色编号。

算术表达式 1 的值应在 0～39 之间，算术表达式 2 的值应在 0～47 之间，但在 40～47 之间时，SCRN 的值将与文本行字符有关。

本命令在高分辨图形方式中无效。

＊在 XMF-BASIC 中，两个算术表达式均可用带 $ 的十六进制数给出。

§ 9.7 HGR 和 HGR_2

命令格式：[行号] HGR

[行号] HGR_2

功能：均可进入高分辨率图形方式，并清除画面屏幕，隐含执行了 HCOLOR＝0。

HGR 只进入第 1 页，且为与文本混合方式，底部留四行文本行，光标置于文本行中。

HGR_2，进入第 2 页，全屏幕为图形方式。

关键字 HGR 和 HGR_2 若用作变量名的开头部分，会先进入图形方式，然后印出

? SYNTAX ERROR

如果程序很大，运行时有可能破坏你的图形，你的图形也可能破坏程序。

使用 HIMEM:8192，可保护图形 1 页，使用 HIMEM:16384，可保护图形 2 页。

§ 9.8 SHG 和 SHG_2

＊这是 XMF-BASIC 增加的命令。

命令格式：[行号] SHG

[行号] SHG_2

功能：SHG 类似于 HGR，但SHG进入的是第 3 图形页。SHG_2 类似于 HGR_2，而 SHG_2 进入的是第 4 图形页。

§ 9.9 HCOLOR=

命令格式：[行号] HCOLOR=算术表达式

功能：用来设定绘制高分辨率图形的颜色编号（参阅第一章 §1.8）。

算术表达式的值应在 0～7 之间。

HCOLOR= 不会被 RUN 改变。且在低分辨率图形方式中无效。

§ 9.10 HPLOT

命令格式：[行号] HPLOT 算术表达式 1，算术表达式 2 [{TO 算术表达式 N1，算术表达式 N2}]

功能：格式中由逗号分隔开的两个算术表达式，代表某坐标点的 X 和 Y 坐标值，表明在高分辨率图形屏幕画点的一个位置。由 TO 连接出的两个算术表达式，又是一个点的坐标位置，TO 表明由刚才画点位置再连线到这个位置。颜色用刚由 HCOLOR=设定的颜色。

逗号左侧算术表达式的值，必须在 0～279 之间；逗号右侧算术表达式的值，必须在 0～191 之间。否则会有出错提示？

?ILLEGAL QUANTITY ERROR

但在混合方式下，逗号右侧的算术表达式的值在 160～191 之间时，没有意义。

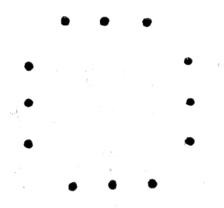
*在 XMF-BASIC 中，算术表达式可用带 $ 的十六进制数给出。

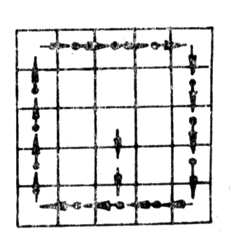
HPLOT 在低分辨率图形方式下无效。

§ 9.11 图形表方法概述

一、图形表的建立

下面左侧一幅图中的 12 个点，排紧后像个〇字。可以设想它是这样画出来的（示意于右侧图中）：设笔的初始位置在正中央，先提笔下移一格，再提笔下移一格，落笔画点后提笔左移一格，落笔画点后再提笔左移一格，再提笔上移一格，落笔画点后再提笔上移一格，……





若把提笔向上、下、左、右方向移动分别用箭头 ↑ ↓ ← → 表示，移动前落笔画点用箭根画一圆点表示，那么，画出上图可用下列箭头序列表示：

← ↓ ←• ←• ↑ •↑ •↑ •↑ → •→ •→ •→ ↓ •↓ •↓ •↓ ← ←•

然后用三位二进制数再来代表这些箭头序列。三位中的左一位，为 1 代表画点（根部有点），为 0 表示不画点；而后两位：00 表示向上移（箭头向上），01 表示向右移，10 表示向下移，11表示向左移。

鉴于本机内存一个单元为二进制 8 位，表示两个箭头只占 6 位（浪费 2 位），而表示 3 个箭头又不够（少一位）。这

里采用了一个特殊方法：第一个箭头占右 3 位（0,1,2 位），第二个箭头占中间 3 位（3,4,5 位），用最左边的两位（6,7 位）来表示第三个箭头的三种情况中的任一个：不画点的向左，不画点的向右和不画点的向下。如果不属于这三种情形，该字节最左边两位填 0，而第三个箭头用后续字节（下一个字节的右三位）表示。如此将未被表示的箭头依次填入后续字节中。当所有箭头表示完后，除用 0 补满最后一个字节外，再将后续字节中 8 位全写成 0，以表示一个图形结束。

用上述方法，可构成一个图形定义。

所举例子的图形定义为：

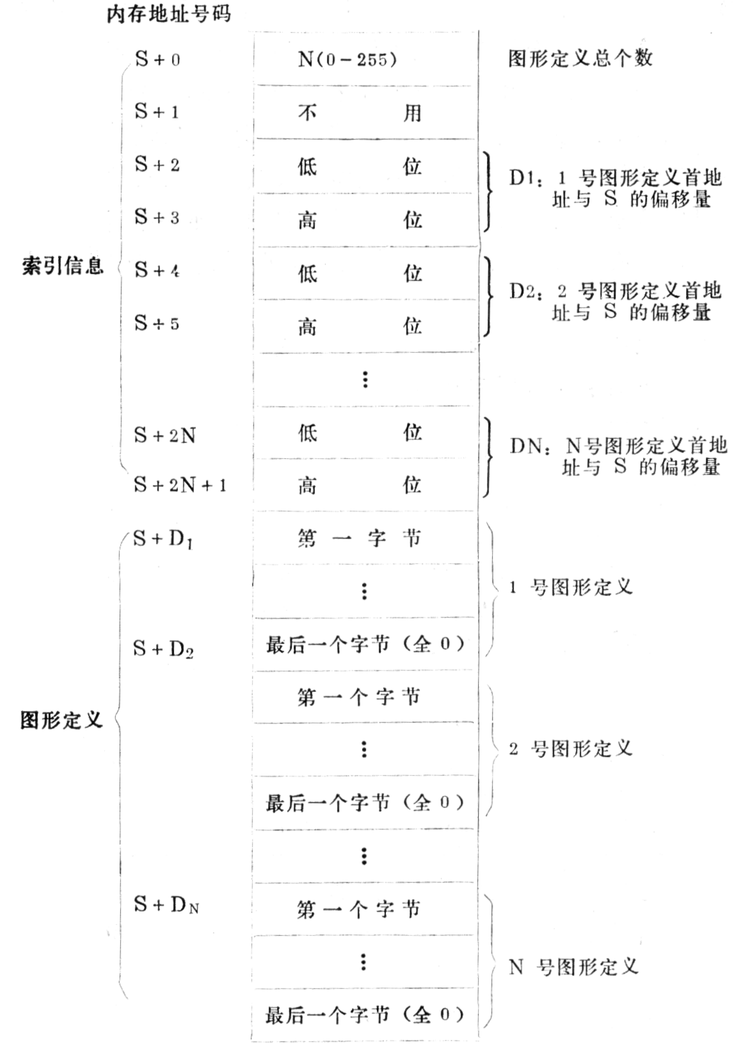
箭　　头	二进制表示	十六进制表示	十进制表示
↓ ↓	00010010	12	18
←• ←•	00111111	3F	63
↑• ↑	00100000	20	32
→ ↑• ↑•	01100100	64	100
•→ •→	00101101	2D	45
↓ •→	00010101	15	21
•↓ •↓	00110110	36	54
← •↓	00011110	1E	30
←•	00000111	07	7
结　　束	00000000	00	0

用上述方法，可构成多个图形定义来。

将若干个图形定义连接在一起，可构造出一个完整的图形表。

构造图形表要确定图形表的初始地址。它要在 RAM 空

间，且不能在所用的高分辨图形区。建议用 24576（即



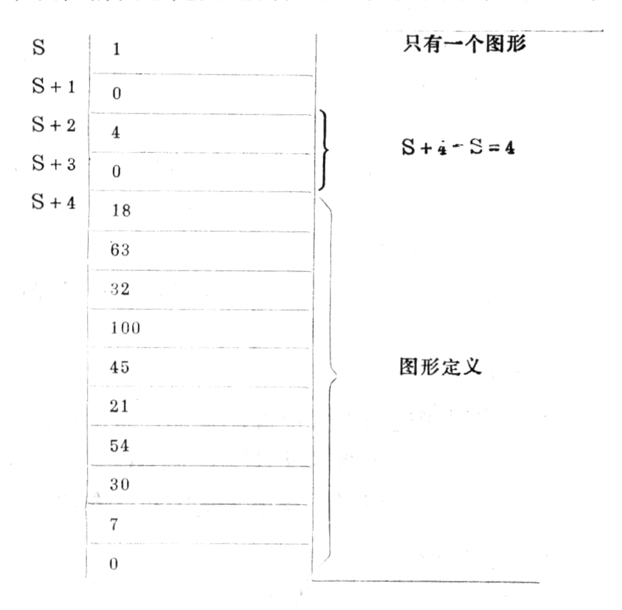
$6000）。

然后把这个地址的高位部分（96，即 $60）计入 233 号单元中，把地址的低位部分（00，即 $00）计入 232 号单元中。这可以用下列命令实现。

```
POKE 232,0
POKE 233,96
```

上页是一个图形表的结构。

对于我们前面已定义过的图形○，其图形表可以是：



二、图形表的送入与保存

前面的叙述，只是纸上谈兵。

用下面的 BASIC 程序，可将上面的图形表送入内存 24576 开始的单元中，以便在电脑中真正建立一张图形表：

```
100 S=24576
110 FOR I=0 TO 13
120 READ A:POKE I+S,A
130 NEXT
140 DATA 1,0,4,0,18,63,32,100,45,21,54,
    30,7,0
150 END
```

用监控命令，也可送入：

```
CALL-151 〈CR〉
  6000:1 0 4 0 12 3F 20 64 2D 15 36 1E 7 0
     〈CR〉
```

在监控状态下，可将上述电脑中的图形表录入磁带：

先用命令

```
0:D 0 〈CR〉
```

将图形表的长度 D（即 13）存入 0 号和 1 号单元中，再用命令

```
0.1 W 6000.600DW 〈CR〉
```

写入磁带（先要接通磁带机电源，置录音状态）。

退出监控状态，可用 CTRL-C 〈CR〉

关于监控命令的使用，在第十章中专门介绍。

三、图形表的使用

图形表在内存建立好，并将始地址计入内存 232 和 233 中之后，运行下列程序可画出一幅图形来：

```
10 HGR
20 HCOLOR=3
30 FOR R=1 TO 50
```

```
40 ROT=R
50 SCALE=R
60 DRAW 1 AT 139,79
70 NEXT
80 END
```

如果想每次画完后暂停一下，并涂去图形的话，可加上下列两行：

```
63 FOR I=0 TO 1000:NEXT I
65 XDRAW 1 AT 139,79
```

程序中用了几个新的命令，将由以下几节说明。

§ 9.12 DRAW 和 XDRAW

命令格式：[行号] DRAW 算术表达式 1[AT 算术表达式 2，算术表达式 3]

[行号] XDRAW 算术表达式 1[AT 算术表达式 2，算术表达式 3]

功能：命令中的算术表达式 1 指明一个图形定义的号码，算术表达式 2 和算术表达式 3 指明在高分辨图形屏幕上点的横坐标和纵坐标。DRAW 的功能是将给定号码的图形在指定的坐标点起笔画出来，用 HCOLOR=所设定过的颜色。

XDRAW 和 DRAW 的功能相同，但取相补的颜色，互补关系是：黑对白，蓝对绿，紫对橙。

因此，用白色先 DRAW 一次，再 XDRAW 一次，会擦掉图形，而不破坏背景。

算术表达式 2 的值，必须在 0～279 之间；算术表达式

3 的值，必须在 0～191 之间。否则会有出错提示

?ILLEGAL QUANTITY ERROR

但在屏幕为混合方式时，底部 4 行总不会显示图形。

当省缺算术表达式 2 和算术表达式 3 时，起笔位置为上次画图时的收笔位置（包括 DRAW、XDRAW 和 HPLOT）。

当电脑中没有图形表，或算术表达式 1 的值超过图形定义的个数，或电脑 232 和 233 号单元的内容未指向图形表的首地址时，那么，DRAW 或 XDRAW 可能画出意想不到的图形，或停顿在那里。

*在 XMF-BASIC 中，三个算术表达式均可用带 $ 的十六进制数给出。

§ 9.13 SCALE=

命令格式：[行号] SCALE= 算术表达式

功能：用来设定图形表画在屏幕上的尺寸。当算术表达式的值为 1 时，图形定义中的一个点在屏幕也是一个点，算术表达式的值为 N 时，图形定义的一个点在屏幕上为 N 个点。

算术表达式的值必须在 0～255 之间，否则会有出错提示：

?ILLEGAL QUANTITY ERROR

当算术表达式的值为 0 时，相当于 SCALE=256。

*在 XMF-BASIC 中，算术表达式可用带 $ 的十六进制数给出。其范围为 $0～$FF。

§9.14 ROT=

命令格式：[行号] ROT= 算术表达式

功能：使图形表产生的图形进行旋转。当算术表达式的值为0时，不旋转；为16时，按顺时针方向转90度；为32时，按顺时针方向转180度……。当SCALE=1时，只有上述4个转角，当SCALE=2时，有8个转角（ROT=8，转45度……）。以此类推。

当算术表达式的值介于两可用值之间时，取较小的一个值。

算术表达式的值，必须在0～255之间。

* 在XMF-BASIC中，算术表达式可用带$的十六进制数给出。其范围为$0～$FF。

§9.15 SHLOAD

命令格式：[行号] SHLOAD

功能：从录音带上读入一个图形表。图形被读入紧贴HIMEM：的区域，并将HIMEM：改设在图形表的首地址，同时把这个地址送入232和233号单元中。因此，图形表读入后可直接使用。

只有CTRL-RESET可以中断SHLOAD。

§9.16 PDL

命令格式：[行号]… PDL（算术表达式）

功能：这是一个函数。算术表达式的值应在0与3之间，用来设定一个游戏摇杆的号码。本函数可得到指定号游戏摇杆的转动控制值。

算术表达式的值在 0～255 之间时，不给出出错提示，但当其值在 4～255 之间时，会产生一些意外。

游戏摇杆的开关，依三个摇杆号码分别对应下面三个单元中的值：

SW0:49249 （－16287,$C061）

SW1:49250 （－16286,$C062）

SW2:49251 （－16285,$C063）

当开关按下时，相应单元的数大于 127，否则小于 127，这可用函数 PEEK 读出来判别是否按下了相应开关。

§ 9.17 TEXT

命令格式：[行号] TEXT

功能：将图形方式屏幕转换成文本方式屏幕，并使光标移至第 21 行开始处。

第十章　监 控 命 令

§ 10.1　简述

监控程序是常驻内存的，它是系统的监控者，又为系统服务。离开它，BASIC 寸步难行。用户若能直接使用它，会大大提高程序效率。通过它，读者还可以进一步分析系统本身。

进入监控状态，可用 CALL-151，或键入 CTRL-SHIFT－－。

＊在 XMF-BASIC 中，也可用 CALL $FF69。

进入监控后，提示符为＊。

退出监控返回 BASIC，可分别用下述命令之一：

CTRL-C 〈CR〉

3D0G 〈CR〉（当操作系统载入电脑时）

CTRL-RESET

CTRL-B 〈CR〉（会清除 BASIC 程序）

在监控状态下，地址和数据都用十六进制数，但输入时一律不用由 $ 导入。输入十六进制数时，左侧的 0 可以省去。

§ 10.2　检查内存的内容

一、命令格式：地址

功能：显示指定地址的内容。

如：

＊E000 〈CR〉

E000-20

指明内存 E000 中的内容为 20（E000 与 20 均为十六进制数。＊是监控提示符）。

二、命令格式：.地址

功能：显示从当前地址开始到指定地址各单元的内容。

三、命令格式：地址 1．地址 2

功能：显示从地址 1 到地址 2 各单元的内容。

四、只敲回车符 〈CR〉

功能：显示从当前地址开始到地址尾数为 7 或 F 的最多 8 个单元的内容。

§ 10.3　修改内存单元的内容

一、命令格式：地址：数

功能：置指定数于指定地址中。

二、命令格式：地址：数 1　数 2　数 3 …数 N。

功能：从指定地址开始，依次置入数 1，数 2，数 3，…数 N。

三、命令格式：:数 1　数 2　数 3 …数 N

功能：从当前地址开始，依次置入数 1，数 2，数 3，…数 N。

§ 10.4　移动一段内存的内容

命令格式：地址 1〈地址 2．地址 3M

功能：将地址 2 到地址 3 的内存中的内容移到从地址 1

开始的一段内存中。

命令执行后，从地址 2 到地址 3 的这段地址如不被从地址 1 开始的那段内存覆盖，其内容不会改变。

当地址 1 大于地址 2 而小于地址 3 时，移动过程中会修改地址 2 到地址 3 的内容，使移动产生错误。

§ 10.5 比较两段内存的内容

命令格式：地址 1〈地址 2．地址 3 V

功能：从地址 1 开始，逐个单元与从地址 2 到地址 3 的内容进行比较，若有不同时，显示出相应地址及不相同的两个数。若两段内存的内容完全一致，什么也不显示（比较完给出提示符 *）。

§ 10.6 将一段内存的内容录入磁带

命令格式：地址 1．地址 2 W

功能：把从地址 1 到地址 2 的内容录入磁带。录入时，要接通录音机电源，置成录音状态。

§ 10.7 从磁带上读一段内容到内存

命令格式：地址 1．地址 2 R

功能：将磁带上的一段内容读入地址 1 到地址 2 的内存中。

§ 10.8 GO 命令

命令格式：地址 G

功能：从指定地址开始，执行机器语言程序。

§ 10.9 LIST 命令

命令格式：地址 L

L

LL…L

功能：从指定地址开始，列出 20 行机器指令及其汇编。

单独一个 L，会从当前地址开始列出 20 行机器指令及其汇编。连续多个 L 命令，每个 L 列出 24 行。

§ 10.10 检查和改变寄存器的内容

命令格式：CTRL-E

:数 1 数 2 数 3 …数 N

功能：CTRL-E 命令，会依次给出 A、X、Y、P、S 各寄存器的内容。

接着用：数 1 数 2 …数 N 命令，会按指定的数依次修改各寄存器的值。其中 N 不得大于 5。

§ 10.11 改变字符显示方式命令

命令格式：I

N

功能：I 命令，置反白方式（类似 BASIC 的 INVERSE 命令）。

N 命令，置正常方式（类似BASIC 的 NORMAL 命令）。

§ 10.12 开关打印机命令

命令格式：1 CTRL-P

0 CTRL-P

功能：1 CTRL-P 接通打印机。0 CTRL-P 关闭打印机。

§ 10.13 强迫转移命令

命令格式：CTRL-Y

功能：相当于执行 3F8。你要在 3F8 中再放一条转移（JMP）指令，使转移到你所需要的程序入口。

§ 10.14 十六进制加减法命令

命令格式：数 1+数 2
数 1-数 2

功能：完成十六进制加减运算。数的范围 0~FF。多于两个字节时只取后两字节。

§ 10.15 多重命令

允许一次敲入多个命令再敲〈CR〉，将依次逐一执行。

§ 10.16 ESC 行编辑功能

在 BASIC 状态下（提示符为]），利用 ESC 键可进行程序行的编辑。这种功能是监控程序提供的。

ESC 编辑功能键有：

ESC A 使光标右移一格
ESC B 使光标左移一格
ESC C 使光标下移一行
ESC D 使光标上移一行

ESC E　从光标处开始清除屏幕至本行尾

ESC F　从光标处开始清除屏幕至屏幕底

ESC @　清除整幅屏幕，光标移至左上角。相当于 HOME 命令。

ESC I　使光标上移一行

ESC J　使光标左移一格

ESC K　使光标右移一格

ESC M　使光标下移一行

后四个 ESC 命令，敲入一次 ESC 后，可连续多次使用 I，J，K，M 中的任一键移动光标，直至敲入别的键，才失去 ESC 控制。

编辑一个程序行时，被编辑的内容必须从头到尾扫描过（如果带行号时，也得扫过）。欲删除的内容用空格键删去，修改或插入的内容通过 ESC 将光标移出来再行敲入，或移至屏上有用部分，用→键扫入。

§ 10.17　控制字符

一、CTRL-RESET

中断任何程序的执行，进入 XMF BASIC，给出] 提示，等待输入。它不破坏程序和变量的当前值。

二、CTRL-B

仅用于监控状态，用它返回 BASIC。它破坏 BASIC 程序。

三、CTRL-C

中断当前程序的执行。在监控状态下，再敲入〈CR〉键，会返回 BASIC。它不破坏程序。

四、CTRL-G

会发出一声“嘟”。

五、CTRL-H

相当于←键。

六、CTRL-U

相当于→键。

七、CTRL-M

相当于 RETURN（即〈CR〉）键。

八、CTRL-J

发出换行信号。

九、CTRL-X

使刚敲入的内容作废。

十、CTRL-S

暂停输出。再敲任一键可恢复输出。

附　　录

附录一　出 错 信 息

一、NEXT WITHOUT FOR

有 NEXT，但没有对应的 FOR。

编号 0（$0）。

二、SYNTAX

语法错。

编号 16($10)

三、RETURN WITHOUT GOSUB

有 RETURN 或 POP，但没有对应的 GOSUB。

编号 22($16)

四、OUT OF DATA

数据个数不够。

编号 42($2A)

五、ILLEGAL QUANTITY

数据超出范围。

编号 53($35)。

六、OVERFLOW

数值太大或太小，溢出。

编号 69($45)。

七、OUT OF MEMORY

内存空间溢出。

编号 77($4D)

八、UNDEF'D STATEMENT

行号不存在（未定义)。

编号 90($5A)。

九、BAD SUBSCRIPT

下标错误。

编号 107($6B)。

十、REDIM'D ARRAY

数组被重复定义。

编号 120($78)。

十一、DIVISION BY ZERO

除数为 0。

编号 133($85)。

十二、ILLEGAL DIRECT

不能为立即执行方式。

编号 149($95)。

十三、TYPE MISMATCH

数据类型不相容。

编号 163($A3)。

十四、STRING TOO LONG

字符串数据太长。

编号 176($B0)。

十五、FORMULA TOO COMPLEX

命令非法。

编号 191($BF)。

十六、CAN'T CONTINUE

不能继续执行。

编号 210($D2)。

十七、UNDEF'D FUNCTION

函数未事先定义。

编号 224($E0)。

附录二　BASIC 关键字内部代码及解释入口（均用十六进数表示）

关键字	内码	解释入口	关键字	内码	解释入口
END	80	D870	HLIN	8E	F232
FOR	81	D766	VLIN	8F	F241
NEXT	82	DCF9	HGR2	90	F3D8
DATA	83	D995	HGR	91	F3E2
INPUT	84	DBB2	HCOLOR=	92	F6E9
DEL	85	F331	HPLOT	93	F6FE
DIM	86	DFD9	DRAW	94	F769
READ	87	DBE2	XDRAW	95	F76F
GR	88	F390	HTAB	96	F7E7
TEXT	89	F399	HOME	97	DO76
PR#	8A	E1E5	ROT=	98	F721
IN#	8B	FIDE	SCALE=	99	F727
CALL	8C	FID5	SHLOAD	9A	F775
PLOT	8D	F225	TRACE	9B	F26D

续表

NOTRACE	9C	F26F	POKE	B9	E77B
NORMAL	9D	F273	PRINT	BA	DAD5
INVERSE	9E	F277	CONT	BB	D896
FLASH	9F	F280	LIST	BC	D6A5
COLOR=	A0	F24F	CLEAR	BD	D66A
POP	A1	D96B	GET	BE	DBA0
VTAB	A2	F256	NEW	BF	D649
HIMEM:	A3	F286	TAB(	C0	DB16
LOMEM:	A4	F2A6	TO	C1	
ONERR	A5	F2CB	FN	C2	E354
RESUME	A6	F318	SPC(	C3	DB16
RECALL	A7	F3BC	THEN	C4	
STORE	A8	F39F	AT	C5	
SPEED=	A9	F262	NOT	C6	DE98
LET	AA	DA46	STEP	C7	
GOTO	AB	D93E	+	C8	E7C1
RUN	AC	D912	−	C9	E7AA
IF	AD	D9C9	*	CA	E982
RESTORE	AE	D849	/	CB	EA69
&	AF	03F5	↑	CC	EE97
GOSUB	B0	D921	AND	CD	DF55
RETURN	B1	D96B			
REM	B2	D9DC	OR	CE	DF4F
STOP	B3	D86E	>	CF	DF65
ON	B4	D9EC	=	D0	DE65
WAIT	B5	E784	<	D1	DE65
LOAD	B6	D8C9	SGN	D2	EB90
SAVE	B7	D8B0	INT	D3	EC23
DEF	B8	E313	ABS	D4	ABAF

续表

关键字	内码	解释入口	关键字	内码	解释入口
USR	D5	000A	VAL	E5	E707
FRE	D6	E2DE	ASC	E6	E6E5
SCRN(	D7	D412	CHR$	E7	E646
PDL	D8	DFCD	LEFT$	E8	E65A
POS	D9	E2FF	RIGHT$	E9	F686
SQR	DA	EE8D	MID$	EA	E691
RND	DB	EFAE	AUTO	EB	E007
LOG	DC	E941	EDIT	EC	D020
EXP	DD	EF09	MAT	ED	F41B
COS	DE	EFEA	@	EE	F41B
SIN	DF	EFF1	SHG2	EF	F1C7
TAN	E0	F03A	SHG	F0	F37A
ATN	E1	F09E	BSAVE	F1	F68F
PEEK	E2	E764	BLOAD	F2	F19E
LEN	E3	E6D6	BRUN	F3	D05B
STR$	E4	E3C5	PCTRL-G	F4	F7F1

附录三　外部命令表与外部信息表的使用

本附录的内容，在 XMF-BASIC 中有效。而在 APPLE Ⅱ BASIC 中是不能实现的。

外部命令是指 BASIC 关键字之外的命令码，它可以用新的英文字词，也可以用汉语拼音，还可以用汉字。用户可以用这些自己定义的外部命令分别替代 BASIC 的某些关键

字或全部关键字。输入或显示，被替代的关键字将以相应的用户定义的命令予以显示，而运行时仍执行原关键字的功能。如全部关键字均以汉字代替，则可使 BASIC 程序将成为完全的汉字 BASIC 程序。

出错信息，也可以由用户定义的信息（如汉语拼音或汉字等）替代部分或全部。

为实现上述功能，用户得先在电脑中构造两张表：外部命令表和外部出错信息表，并将零页的 $EB、$EC 置入相应的数字，使其指向第一张表（外部命令表）的首地址。$EB 存放这个地址的低位，$EC 存放这个地址的高位。

举例来说，假如想从内存的 $7600（30208）开始构造外部命令表，可通过下述两个命令将这个首地址送入 $EB，$EC 中：

POKE $EB，0： POKE $EC，$76

这两个命令相当于

POKE 235，0： POKE 236：118

当然，还可以在进入监控状态后，用下述命令来完成送首址的任务：

EB:0 76

接着，我们假定想用汉字的“清”字替代 HOME，还想用汉字的“到”字替代关键字 TO。“清”字国标码为 $47 和 $65，“到”的国标码为 $35 和 $3D，而 HOME 的内码为 $97，TO 的内码为 $C1。

这时，我们可以从 $7600 开始造外部命令表，如下页表：

开始两个字节给出一个地址，表明外部出错信息表的首地址（上表中为 $7700）。如果没有外部出错信息表，则这

地址码	内容	意义	
7600	00	指向外部出错信息表（如无外部出错信息表，这两个字节全为 00）	
7601	77		
7602	04	长度	一个命令
7603	FF	国标码引导符	
7604	47	“清”的国标码	
7605	65		
7606	97	HOME 的内码	
7607	04	长度	一个命令
7608	FF	国标码引导符	
7609	35	“到”的国标码	
760A	3D		
760B	C1	OT 的内码	
760C	00	表的结束标志	

两个字节中都填入 $00。

第三个字节到第 7 个字节（$7602～$7606）中为 HOME 的对照信息。其中首字节给出对照信息的长度为 4（$7603～$7606），最后一个字节为关键字代码，其余各字节为外部命令码（如想用汉字拼音 QING 替代 HOME，显然长度为 5，而 $7602～$7607 中依次应为 05 D1 C9 CE C7 97。而第二个命令的信息向后顺移两个字节）。

上表中 $7607～$760B 为第二个命令的全部信息。

最后一个字节中的 00，表明这张外部命令表的结束。

上述外部命令表，在监控状态下可以很方便的输入电脑中：

```
7600:0 77 4 FF 47 65 97 4 FF 35 3D C1 0
```

外部出错信息表构造稍为复杂，它的首地址总是由外部命令表的开始两个字节定义的。

外部出错信息表开始六个字节，是相对本出错信息表首址的偏移量，分别指明下列六个信息的外部信息的首地址：

第一个字节：存放有编号信息（即附录一所给出的出错信息）首地址相对于本表首址的偏移量；

第二个字节：存放相应于英文 ERROR“嘟”的外部信息首址与本表首址的偏移量；

第三个字节：存放相应于英文 IN 的外部信息首址与本表首址的偏移量；

第四个字节：存放相应于英文 CTRL-M BREAK 的外部信息首址与本表首址的偏移量；

第五个字节：存放相应于英文？EXTRA IGNORED 的外部信息首址与本表首址的偏移量；

第六个字节：存放相应于英文？REENTRE 的外部信息首址与本表首址的偏移量。

如果上述某一个（或一类）信息没有定义相应的外部信息，则在上述相应字节中填入 0。

本表的结束用全 1 字节（即填入 $FF）作标志。

下面举例从 $7700 开始构造一张外部出错信息表，希望用汉语拼音 JU FA（句法）CUO（错）来替代英文 SYNTAX ERROR 信息。见下页表。

此表造出后，如遇句法错误，将不提示 SYNTAX ERROR，而提示 JU FA CUO（句法错）。

如果需增加别的有编号的外部出错信息，可用表的最后一个地址（放 FF 的地方）用类似方法继续造，造完后结尾

地址	内容	意义
7700	0A	有编号出错信息首地址：7700+0A=770A
7701	06	相应于英文 ERROR"嘟"的外部信息首地址： 7700+06=7706
7702	00	无对应英文 IN 的外部信息
7703	00	无对应英文 CTRL-M BREAK 的外部信息
7704	00	无对应英文？EXTRA IGNORED 的外部信息
7705	00	无对应英文？REENTER 的外部信息
7706	03	长度（外部信息，不含本字节） ┐ 一个无编号信息
7707	C3	C 的ASCII 码 ┐ CUO（错）信息
7708	D5	U 的ASCII 码
7709	CF	O 的ASCII 码 ┘
770A	10	SYNTAX 的编号
770B	05	外部信息长度（含本字节） ┐ 一个有编号的信息
770C	CA	J 的ASCII 码 ┐ JUFA（句法）信息
770D	D5	U 的ASCII 码
770E	C6	F 的ASCII 码
770F	C1	A 的ASCII 码 ┘
7710	FF	本表结束

放一个全 1 字节。

如果需要造另外的无编号外部信息，可插入有编号信息之前(将有编号信息向高地址区移动，并修改本表首字节的内容)，同时还要修改前 6 个字节中相应字节的数字(偏移量)。

附录四 常用子程序入口与专用单元

除附录二所列解释入口外，这里再举出一些常用入口。其中有些入口不能直接用 CALL 调用，但可供分析参考。

-8192（$E000）在没有操作系统 DOS 时，负责内存初始化的工作。这段程序，会破坏 DOS。

-11204（$D43C）暖启动入口，会给出提示符“]”，然后等待用户输入。

-65359（$B1）BASIC 程序运行时扫描程序入口。

-11246（$D412）输出出错信息的程序入口。

-10962（$D52E）接收键入一行 BASIC 程序或命令，并对键盘输入缓冲区作初步处理。

-10919（$D559）将键盘输入缓冲区的 BASIC 关键字置换成相应内部代码。

-11159（$D469）将带行号的一个程序行移入程序区相应位置。（在 APPLE Ⅱ BASIC 中，此入口为 -11172 即 $D45C）。

-10286（$D7D2）一个程序执行结束后，转至此入口继续执行后继续程序或停机。

-10205（$D823）这里是一条转移指令，指向 -10286（$D7D2）。

-10200（$D828）分析一个关键字并转入相应的解释入口。

-6408（$E6F8）将取值范围只能在 0～255 的算术表达式的值转换成二进制数，存入变址寄存器 X 中，并将后续字节内容扫入累加器 A 中。

-5039（$EC51）将算术表达式中的常数，转换成浮点二进制数置入浮点累加器中。在 APPLE Ⅱ BASIC 中，此入口为 -5046（$EC4A）。

-4812（$ED34）在堆栈区组装输出内容的显示 ASCII 码。

－6700（$E5D4）向字符串数据区送字符串数据。

－151（$FF69）进入监控状态入口。

－662（$FD6A）接收一个程序行的输入。

－531（$FDED）输出一个字符。

－528（$FDFO）输出一个字符到屏幕。

－636（$FD84）产生一 RETURN 字符。

－1720（$F948）输出 3 个空格。

十六进制地址	十进制地址		用途	
	正地址	负地址		
$C050	49232	－16304	图形方式	
$C051	49233	－16303	文本方式	
$C052	49234	－16302	整幅屏幕	
$C053	49235	－16301	混合方式	
$C054	49236	－16300	第 1 页或第 3 页	
$C055	49237	－16299	第 2 页或第 4 页	
$C056	49238	－16298	低分辨率图形	
$C057	49239	－16297	高分辨率图形	
$C000	49152	－16384	键盘接收的单元	
$C010	49168	－16368	清键单元	
$C030	49200	－16336	相声器	
$C061	49249	－16287	0 号按钮开关	游戏摇桿
$C062	49250	－16286	1 号按钮开关	
$C063	49251	－16285	2 号按钮开关	
$C064	49252	－16284	0 号旋转值	
$C065	49253	－16283	1 号旋转值	
$C066	49254	－16282	2 号旋转值	
$C067	49255	－16281	3 号旋转值	

-198（$FF3A）输出一个 CTRL-G（产生“嘟”）的程序入口。

-715（$FD35）获得一个字符的程序入口。

-741（$FD1B）读键盘程序入口。

-856（$FCA8）延时程序入口。

常用专用单元如上页表（0 页单元见附录六）。

附录五 内存分配及 DOS 与图形第 4 页的地址冲突处理

小蜜蜂机的 RAM 为 64KB。其内存分配在低地址区($0000～$5FFF)与 APPLEⅡ 完全一样，高地址区却有些区别，如下页图。

图形中有些区域分左右两部分或左中右三部分，表明它们都可以进入。但通常以后占者破坏先占者。

由于 DOS 及其预留的文件缓冲区占用内存 $9600～$BFFF，因此，XMF-1 机一旦载入 DOS，高分辨率图形第 4 页将不能使用，汉字屏幕也不得使用这一图形页。这就是内存地址的冲突。另一方面，由于小蜜蜂机的内存已扩展到 64K，扩展的 16K 又在那里置闲。因此，小蜜蜂采用了把 DOS 移入扩展 16K 中的办法（实际上只用了其中 12K，即上图中 RAM 中最高地址标有用户区的 12K），另留出 RAM 中 $BF00～$BFFF（对于 DOS3.3）或 $BE00～$BFFF（对 DAVID-DOS）作为界面。这样第四图形页就可以使用，并扩大了用户可使用的区域。

RAM

地址	内容		
$0000	系统 0 页		
$0100	系统堆栈区		
$0200	键入缓冲区		
$0300	用户机器语言程序		
$03D0	DOS向量		
$0400	低分辨率图形 1 页	文本 1 页	
$0800	低分辨率图形 2 页	文本 2 页	
$0C00			
$2000	高分辨率图形 1 页		用户区
$4000	高分辨率图形 2 页		
$6000	高分辨率图形 3 页		
$8000	高分辨率图形 4 页	汉字屏幕	
$9600			
$A000	DOS		
$C000	I/O 区		
$D000	用户区		
$FFFF			

ROM

BASIC
解释程序 1
监控程序

ROM

BASIC
解释程序 2

鉴于汉字屏幕用第四图形页，因此只要进入汉字系统，它会自动将 DOS 移走。如果不使用汉字系统，用户可直接执行命令 CALL$77A 或命令 CALL$7FA；对于DAVID-DOS，还可执行命令 HIDOS。这样也可以将 DOS 移走。

移入扩展 RAM 中的 DOS，除个别命令（如 DOS3.3 中的 INIT，DAVID-DOS 中的 HIDOS）外，别的DOS 命令都能照常执行。

附录六　关于零页用法

系统在零页（$0000～$00FF）中设置了许多指针、数值、标志、计数器、累加器。用户不可轻易修改它们。了解它们，却很重要。这里仅主要用法列示如下：

00～02	放一条 JMP $D43C 的指令，转 BAISC 暖启动入口，它不破坏程序和数据。 其中内容被修改后，要注意恢复。
03～05	放一条 JMP $DB3B 的指令（APPLE Ⅱ 放的是 JMP $DB3A）。可印出一个字符串。
06～09	没用。
0A～0C	放一条跳转指令，为函数 USR 的入口。
0D 和 0E	用于扫描字符串及 REM 内容等的开始和结束标志。
0F	工作单元。常用于计数。如统计一个程序行或字符串的长度，在把关键字转换为内码时记代号和记数组的维数等。
10	工作单元。遇 DIM 时，置入数组名首字符的

	ASCII 码，用来检查一个数组是否重复定义。
11	标志单元。在计算一个表达式时，如系字符串型变量，置FF；如系算术型量，置 0。
12	标志单元。整型数，置 80；实型数，置 0。
13	标志单元。处理键盘输入时，遇冒号，置 0；遇 DATA，置 49；其它置入 4。
14	标志单元。标志是否允许用整型数。允许时置 0；不允许时置 80。
15	标志单元。用于 INPUT，READ 和 GET 执行中，遇 GET 时置 40，遇 INPUT 时置 0，遇 READ 时置 98。
16	标志单元。在处理关系表达式时，存放关系运算符号代码。
1A～1B	高分辨率图形清屏时，放图形区首址，清完一行后，1B 中的数加 1，直至清完全屏幕。用图形表作图时，1A～1B 先被置入图形表首址。
1C	画高分辨图形时，置入颜色标志。
20	文本窗口左侧位置。
21	文本窗口宽（字符个数）。
22	文本窗口顶行位置。
23	文本窗口末行位置。
24	光标所在水平位置（左起第几个字符位置）。
25	光标所在垂直位置（第几行）。
26～27	高分辨率画图时的寄存器，放该行最左一个字节的地址。
28～29	光标所在行最左一个字符在屏幕存储器的地址。

2A～2B　屏幕滚动输出时，记滚动行最左一个字符在屏幕存储器的地址。在 XMF-BASIC 中，处理自定义函数时，作临时寄存器。

2C　低分辨率画水平线时终止位置。

2D　低分辨率画垂直线时终止位置。

2E　控制在低分辨率图形屏幕画色块时用左4位（下色块，此时 2E 中为 F0），还是用右4位（上色块，此时 2E 中为 0F），还是全用（上下色块全画，此时 2E 中为 FF）。

2F　录音带输入时，用以检查波形有无改变。

30　低分辨率绘图时，颜色寄存器。

31　分析监控命令时所用的寄存器。

32　字符显示方式控制器：

32 中为 FF 时，正常方式；

32 中为 7F 时，反白方式；

32 中为 3F 时，闪烁方式。

33　存放提示符的 ASCII 码：

33 中为 DD，提示“]”；

33 中为 AA，提示“*”；

33 中为 BE，提示“>”；

33 中为 A1，提示“!”。

34 或 35　暂存变址寄存器 Y 的值。

36～37　输出向量，跳到输出程序段。

38～39　输入向量，跳到输入程序段。

3A～3B　监控程序用。通常置后续命令的地址。

3C～3D　读写磁带时的首地址。在 XMF-BASIC 中，

这两个单元用来记录自定义函数参数个数和作辅助寄存器用。

3E～3F	读写磁带的终结地址
40～41	临时寄存器。
42～43	临时寄存器。
44～45	临时寄存器。
46	X 临时寄存器。
47	Y 临时寄存器。
48	P 临时寄存器。
49	S 临时寄存器。
4A～4D	没用。
4E～4F	放着一个0～65535的随机数。在XMF-BASIC中也用作临时寄存器。
50～51	高级语言中给出的数需换成十六进制数时（如十进制地址换成十六进制地址，十进制程序行号换成十六进制代码等）放被转换后的十六进制数。
52～5D	多用途指针。
5E～5F	经常放被转移的地址。
60～61	处理关系表达式时存放数据。修改程序区内容时，借助用作被移动位置地址。
62～66	存放上次进行浮点数乘除的结果。
67～68	指向 BASIC 程序区的首地址。
69～6A	指向简单变量表的首地址。即 LOMEM: 值。
6B～6C	指向数组表的首地址。
6D～6E	指向数组表的末地址。
6F～70	指向字符串数据区的首地址。

71～72	当前字符串首址。
73～74	存放 HIMEM：的值。
75～76	正在执行的程序的行号。 76 中为 FF 时，为立即执行方式；否则为程序执行方式。
77～78	上一个被执行的程序行的行号。
79～7A	指向下一个被执行语句首字节在程序区的地址。
7B～7C	DATA 已读到的行号。
7D～7E	指向下一个欲读数据在程序区首地址。
7F～80	INPUT 获得数据的地址；READ 正在读的数据的地址。
81～82	刚刚用到的变量名。
83～84	正在处理的一个变量的数据首地址。
85～86	为一个变量存入数据时的首地址。
87	记录当前处理的算术运算符号（除关系运算符外）的一个代码。
89	存放正在处理的关系运算符的一个代码。
8A～8B	函数数据首地址。
8F	打扫内存空间时，作指针用。
90～92	放一条 JMP 指令，指向被处理函数的相应入口。
93～97	浮点数暂存器 1。
98～9C	浮点数暂存器 2。
9D～A3	浮点数累加器。
A4	浮点数运算时暂存器。用于浮点数运算。
A5～AA	浮点数暂存器。用于浮点数运算。

AB～AC　在移动字符串子程序中，指向该字符串首地址。

AD～AE　在处理字符串时，指向该字符串首地址。

AF～B0　指向 BASIC 程序的最后一个字节（程序尾指针）。

B1～C8　扫描程序区或扫描键盘缓冲区时的扫描子程序（用以读出下一个字节的内容）。

C9～CD　记录上一次 RND 产生的随机数。

D0～D5　绘制高分辨率图形时工作单元。

D2～D3　产生错误时，该行程序行号在程序区的首地址。

D6　一个神密的寄存器：

如果 D6 中的数 >=80，敲入任何命令都当作敲入 RUN 命令。

如果 D6 的数 <80，一切正常。

D8　关于出错处理标志：

当 D8 中的数 >=80 时，如有 ON ERR GOTO 的情形出错转相应行号去处理；

当 D8 中的数 <80 时，一遇错误就给出出错信息并停机。

DA～DB　出现错误时，记录出错程序行行号。

DC～DD　出现错误时，存放扫描指针位置。

DE　存放出错的编号。

DF　出现错误时，放 S｛堆栈指针）值。

E0～E1　记高分辨率绘图时点的水平位置。

E2　记高分辨率绘图时点的垂直位置。

E4　记高分辨率绘图时颜色代号。

E5　记高分辨率绘图时点所在的字节号(从左端算起)。

E6	高分辨率绘图时页面存储区高位字节：第 1 页记 20；第 2 页记 40；第 3 页记 60；第 4 页记 80。
E7	记 SCALE 的值，图形放大尺寸。
E8～E9	记图形表首址。
EA	高分辨率作图时的碰撞计数器。
EB～EC	XMF-BASIC 中记外部命令表首地址。若无外部命令表和外部出错信息，此二单元填 00。
ED	标志单元： 汉字方式下为 FF，西文系统下为 00。
F0	低分辨率图形使用的暂存器。
F1	记 SPEED= 的值，控制字符输出速度。
F4～F7	ON ERR 指针。
F8	存放 S 值。
F9	存放 ROT 的值，控制图形旋转角度。

附录七　机内 ASCII 码

第一章已列出了键盘 ASCII 码和显示 ASCII 码。键盘 ASCII 码指敲键时从键盘接收单元收到的码；显示 ASCII 是指输出显示时用的编码；而机内 ASCII 码指字符在电脑中存放时的编码。所有字符串运算，字符串函数都使用机内编码。

机内 ASCII 码如下：

位号 6	0	0	0	0	1	1	1	1
5	0	0	1	1	0	0	1	1
3 2 1 0 4	0	1	0	1	0	1	0	1
0 0 0 0	NUL	DLE	SP	0	@	P		p
0 0 0 1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	STX	DC2	″	2	B	R	b	r
0 0 1 1	ETX	DC3	#	3	C	S	c	s
0 1 0 0	EOT	DC4	$	4	D	T	d	t
0 1 0 1	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	ACK	SYN	&	6	F	V	f	v
0 1 1 1	BEL	ETB	′	7	G	W	g	w
1 0 0 0	BS	CAN	(	8	H	X	h	x
1 0 0 1	HT	EM	)	9	I	Y	i	y
1 0 1 0	LF	SUB	*	:	J	Z	j	z
1 0 1 1	VT	ESC	+	;	K	[	k	{
1 1 0 0	FF	FS	,	<	L	¥	l	|
1 1 0 1	CR	GS	–	=	M	]	m	}
1 1 1 0	SO	RS	.	>	N	∧	n	~
1 1 1 1	SI	US	/	?	O	—	o	DEL

此表内的编码，是由键盘 ASCII 码高位变为 0 转化来的。

第二部分 XMF-I汉字系统使用说明

第十一章 汉字系统概述

“小蜜蜂-I”微型计算机为用户提供了一个强有力的汉字系统，而掌握它的使用方法又甚为容易。它支持 XMF-BASIC，主要功能有：

1. 可显示和打印国际 GB2312-80 的一、二级全部简体汉字 6763 个和绝大部分繁体汉字字形，其中显示字形为 16×16 点阵，打印字形为 16×16、24×24、32×32点阵。

2. 支持拼音、全拼、部首、国标、区位五种汉字输入方式，并具有词组输入功能，汉字和 ASCII 码可混合输入。

（1）“拼音”为容错式的拼音输入方法，根据汉语拼音符和英文字母键的一一对应关系输入汉字，每输入一个字母，汉字系统立即搜寻相应汉字，不断输入字母，则不断在提示行显示对应汉字，直到该汉字拼音字母输入完。提示行中所显示的汉字为该字的同音字，每次只能显示 6 个汉字，可按空格键（在汉字系统中赋于其一个特殊名称：卷帘键）寻找到所需汉字。拼音输入法比较方便，与拼音习惯用法一

致，只要懂得汉语拼音的用户便可立即使用。

（2）“全拼”为非容错式拼音输入方法，是专为儿童学习汉语拼音而设计的。根据汉语拼音输入字母寻找汉字，这与“拼音”输入法一致，但是，只有当所要求显示的汉语拼音字母全部送完，而且是正确的，才能在提示行中显示出该汉字或同音字。

（3）“部首”输入方式是一种形码输入方法。

（4）“区位”和“国标”输入完全符合 GB2312-80 编码。

（5）为使汉字输入得更快，“全拼”、“拼音”和“部首”输入方法还支持 7000 个联想式词组输入。

（6）对于常用的表格符，除了借助于国标码和区位码之外，还提供了一种简便的输入方法。

3. 支持九针打印机，可打印 3 种长方字形，3 种正方字形，2 种左旋 90° 字形。

4. 具有很强的图形功能，它通过在屏幕上开辟一个可调大小的文本窗口和一个可调大小的图形窗口实现汉字和图形的混合显示。

第十二章 汉 字 输 入

§ 12.1 启动汉字系统

在“]”提示符下打入 PR#5，便进入汉字系统。这时，屏幕左下角应出现“西文”二字，“西文”二字上方有一个“]”符号为提示符。

§ 12.2 标志字、提示行、乒乓键

一、标志字：当汉字系统引导完成后，屏幕左下角的“西文”二字，称为输入方式标志字，简称标志字。它指出当前的输入方式。“小蜜蜂”汉字共有 6 种标志字，它们是：西文、拼音、全拼、部首、国标、区位。

二、提示行：标志字所在的那一行称为屏幕提示行，在“拼音”、“全拼”和“部首”输入方式下，它一次可出现

乒 乓 键	状态（标志字）1	状态（标志字）2
CTRL-N	西 文	拼 音
CTRL-K	西 文	全 拼
CTRL-W	西 文	部 首
CTRL-F	西 文	国 标
CTRL-Z	西 文	区 位
CTRL-L	（汉字简、繁体转换，无标志字）	
CTRL-Q	（表格符输入，无标志字）	

6个带序号的同音字或同形字供用户选择。

三、乒乓键：不同的标志字代表不同的输入方式，这些输入方式之间的转换是通过5个乒乓键实现的，另有两个乒乓键为汉字简、繁体字形及表格符的转换键，但无标志字提示。

如上表：

例1：键入 CTRL-N，标志字为“西文”。

再键入 CTRL-N，标志字为“拼音”。

再键入 CTRL-N，标志字为“西文”。

再键入 CTRL-N，标志字为“拼音”。

反复键入 CTRL-N 多次，标志字总是在“西文”和“拼音”两种状态下切换，我们把具有这种功能的键叫做“乒乓键”。

上述乒乓键都是两态切换键，但 CTRL-L，CTRL-Q 在切换过程中标志字没有改变。

§ 12.3 “拼音”及“全拼”输入

“拼音”及“全拼”输入规则和顺序是：

一、用 CTRL-N 进入“拼音”输入方式，标志字为“拼音”；用 CTRL-K 进入“全拼”输入方式，标志字为“全拼”。

二、根据拼音符和英文字符键一一对应关系输入汉字。

例2：“小 Xiao”字的键盘输入为 XIAO（注意：输入只能用大写字母，以下同）。

三、提示行上出现6个同音字，可供选择，如果需要的字不在提示行上，按一下空格键（又称同音字卷帘键）查询下一提示行。

例3："小 Xiao"的拼音 XIAO 输入后，提示行为：

拼音 XIAO 1萧 2硝 3霄 4削 5哮 6嚣

它们都是"小"的同音字，但"小"字不在提示行上，按空格键，提示行变为：

拼音 XIAO 1销 2消 3宵 4淆 5晓 6小

"小"字出现于提示行，序号为6。

四、按照提示行所标明的数字输入所需要的汉字序号。

例4："小 Xiao"的序号为6，按一数字键"6"，"小"字便出现于"]"提示符之后的光标"-"所在位置，这样，"小"字的输入就完成了。

五、如果想再送一个汉字，只需在以上汉字送完之后，直接打下一个汉字的拼音即可，不需另外的转换。

例5："小 Xiao"送完后，想送"博 bo"字，则直接打 BO，再打空格键，再打字符键"1"。

六、如果输入完汉字后还想送英文，则按 CTRL-N 回到"西文"，就可以输入标准英文字母了，再按 CTRL-N 还可以回到"拼音"，这样，在同一程序行上中、西文可以混合输入。

七、要结束拼音输入，按 CTRL-N 回到"西文"。

八、特殊技巧

(1) 复制功能：提示行上现行显示的汉字可以多次复制，卷帘键（即空格键）可重复卷帘。

例6：在"拼音"标志字下键入 XIAO，把"萧"字的序号1连按4次，然后把"哮"的序号5连按6次，程序行上连续出现4个"萧"和6个"哮"，再按空格键十几

次，提示行将逐行重复出现“XIAO”的所有同音字。

但是当所显示的汉字带有词组时，该汉字打上屏幕后，提示行的现行状态改变，此时即失去对原汉字的重复功能。

例7：在“拼音”标志下，键入“XIAO”，提示行显示出：

1萧 2硝 3霄 4削 5哮 6嚣

按“6”键，“嚣”字被打上屏幕显示，同时提示行状态改变为：

1张

即联想词组“嚣张”，此时失去对“嚣”字的重复功能，卷帘键也失去对“XIAO”的同音字的重复显示功能。

(2) 简易输入：对于一些常用字，有时一个字的拼音未输完，这个字已出现在提示行上，此时可直接按这个字的序号，而不必等到输完拼音以后再选字。如“兄 Xiong”，输完 XIO 后，“兄”字已出现于提示行，这时按下“1”键就可以了。当然要是输完 XIONG 再按“1”键也有同样效果，只不过要多敲几下键。

(3) 退格键“←”可清除提示行。

例8：要输入“波 bo”，但不慎打了 BA，这时你可以用“←”键清掉提示行，重新输入正确的“BO”。

例9：打印字符串“小博士汉字”

程序：10PRINT“小博士汉字”

操作步骤：(注意：在键入下列字符时，双引“ ”的内容要打入，但双引号本身不要打入，破折号“——”指出操作顺序，也不要打入，[SP]代表空格键，如3[SP]则代表按3次空格键，以此类推。以后各例均如此。)

"CTRL-N" 进入西文——"10 PRINT"——"CTRL-N" 进入拼音

"XIAO" —— "2[SP]" —— "6" ；小
"BO" —— "2[SP]" —— "1" ；博
"SHI" —— "4[SP]" —— "2" ；士
"HAN" —— "3[SP]" —— "1" ；汉
—— "1" ；字

〈回车〉

"CTRL-N" 进入西文

"RUN"〈回车〉

§ 12.4 "部首"输入

偏旁部首编码，简称部首码，是按照偏旁部首拆分的编码。根据字形结构，把构字单元分成 26 组，分别对应标准键盘的 26 个字母。对应关系见附录二中的附表一。

部首码的编码规则和书写规则相符，另外加上部分补充规则，组成部首码的编码规则。

部首码编码规则：

一、把字拆分成编码构件，按照书写顺序逐一编码。

二、每组构件分主从两部分，主构件不能拆分；一组构件内以主构件加上从构件可以得到的构件，不能用其它两组构件拼接。

对上述规则的说明：

(1) 对字进行编码，以书写笔顺为次序，取字的一、二、三、末 4 个构件；超过 4 个构件，只取一、二、三、末，不足 4 个，有几个取几个。例如：

藻：艹氵口口口木

有6码，只取艹氵口木就够了，如果多取了，多取的码子无效。

华：亻七十　只有三码，全部取下就行了。

（2）取码时，一个构件的第一笔在前，则取整个构件在前。例如：

国：口玉

栽：戈木

“国”，构件“口”第一笔为竖，最后一笔为横，取码时，把“口”放首码。“栽”的取码原则也是这样。

（3）有些汉字在拆分时，不具有明显的唯一性，则在拆分时，构件的分割，先输入的部分从多不从少。

例如：兰：䒑一一

失：𠂉一人

构件中有“丷䒑，𠂉𠂉”，先输入部分取构件笔划多的，如上面取法，而不取“丷一一一”或“一一人”

（4）编码规则第二条的说明：

R 组构件的主构件是“门”，用从构件加主构件可以得到“月”、“用”构件，那么，在遇到汉字中“月”、“用”构件时，就不能用“门”加“—”加“—”拼成“月”，不能用“门”加“キ”拼成“用”代替之，其余亦然。

（5）有少量偏旁部首拆分时，规定只取首末码，详见附录二的附表二。

（6）为提高输入速度，对使用频度较高的“的、是、和、在、有、着、要、地、得”九个字，除了可以按正常规则编码，同时给以简码“C”。

（7）附录二内附表三列出了编码旁部首编码例字，凡由

这些例字组成的合体字，均可按例字拆法进行编码。

例如：“尚”，“鬼”的编码为：

尚：小门口（MRG）

鬼：丿田厶（EWO）

对于合体的“敞”、“槐”则为：

敞：小门攵（MRN）

槐：木丿厶（CEO）

（8）附表三中的“△”，表示可以任意输入或省略。

§ 12.5 “国标”输入

国标 GB2312-80 规定了 3755 个一级汉字和 3008 个二级汉字，每个汉字对应一个 4 位十六进制码，输入这个码，对应汉字就在屏幕上出现。如果输入的国标码不合法，系统会以响铃警告。如果要用国标码输入汉字，手头要有一本 GB2312-80 码本。事实上，国标码输入方法是很少被采用的，但汉字在计算机内部是用国标码表示的。

例 10：按 CTRL-F 进入“国标”

键入代码		相应汉字
5021	——	小
3229	——	博
4A3F	——	士
3A3A	——	汉
5756	——	字
7821	——	响铃（非法输入）

§ 12.6 “区位”输入

区位码与国标码同为国标 GB2312-80 规定的码本，用4位十进制数表示一个汉字，高两位代表“区”号，低两位代表“位”号，区位码即由此得名，这样共有 99 个区，每区 99 个位，即每区有 99 个汉字。

例 11：按 CTRL-Z 进入“区位”方式

键入代码		相应汉字
4801	——	小
1809	——	博
4231	——	士
2626	——	汉
5554	——	字
8801	——	响铃，非法输入

§ 12.7 简、繁体字形转换

CTRL-L 是一个实现汉字简、繁体转换的乒乓键，它没有标志字注明当前状态。

CTRL-L 键在标志字处于“拼音”、“全拼”、“部首”、“国标”、“区位”状态时有效，以“拼音”为例：

例 12：输入“声 Sheng”

“CTRL-N”进入拼音，打入“SHENG”——“1”，出现简体“声”

打“CTRL-L”进入繁体，打入“SHENG”——“1”，出现繁体“聲”，连打几个“1”便出现几个繁体“ ”。

（注意：提示行上仍为简体，但输入程序行时已变为繁体）

"CTRL-L" 回到简体，打入"SHENG"——"1"，出现简体"声"

"CTRL-L" 进入繁体，打入"SHENG"——"1"，出现繁体"聲"

"CTRL-L" 回到简体。

"部首"、"国标"、"区位"几种状态下同样利用 CTRL-L 作为简、繁体的切换输入键。由上例可以看出，当按 CTRL-L 进入繁体时，标志字无变化，无论CTRL-L 如何变化，都不改变当前的中文输入方式。

§ 12.8 联想式词组

对于"拼音"、"全拼"和"部首"三种输入方法，"小蜜蜂"汉字提供了 7000 个常用词组，可使输入更加简单和快速。如果一个汉字具有连带词组，那么在输完这个汉字以后，词组便自动出现于提示行上，不需事先记忆。

例 13：输入词组"小说"

CTRL-N 进入"拼音"，送完"小"字以后，提示行为：

1姐 2说 3型 4组 5孩 6时

按数字键"2"，把"说 SHUO"字送上去了。

如果要送词组"小心翼翼"，那么在"小"字送完后，忽略第一个词组提示行，按空格键，第二个提示行为：

1朋 2友 3学 4生 5心 6翼

连打数字键 5、6、6，"心翼翼"送上去了。

由此可见，"小蜜蜂"汉字的词组功能对用户来讲是十分方便的，除了不用打它的拼音码或部首码以外，其它规则

都与当前所处的汉字输入方式一样。但是，由于内部空间有限，并不是每个汉字都有联想词组，已固化的词组是根据词频统计成果表确定的。

§ 12.9 “表格符”输入方法

造表是用户常遇到的问题，但用国标码或区位码输入表格符很麻烦，“小蜜蜂”提供了一种较为方便的方法。

当标志字为“拼音”、“全拼”或“部首”时，只要按一下乒乓键 CTRL-Q，最常用的十个表格符便出现于提示行上，第一次出现五个，按空格键出现下五个，反复按空格键，表格符循环出现。

键入相应序号输入所需表格符，当表格符输完后，再按 CTRL-Q 便回到原来的输入状态。

第十三章 汉字输出

"小蜜蜂"汉字可支持九针打印机，如 CP-80，FX-100，FAX-100 等型号的打印机，可打印 8 种字形。

§ 13.1 打印机启动

一、键盘启动：乒乓键 CTRL-P 可启动或关闭打印机，这种方式适用于程序的列表或调试等。当 CTRL-P 启动打印机时，屏幕上显示的内容全被打印机所拷贝。

二、程序启动：POKE40960,N（16 进制 $A000。N 为非零数时，启动打印机，N 为零，关闭打印机）。

例 1：
```
10 POKE40960,2          （启动打印机）
20 PRINT "小蜜蜂汉字"
30 POKE40960,0          （关闭打印机）
40 PRINT "小蜜蜂汉字"
RUN〈回车〉
```

显示器上出现两行"小蜜蜂汉字"，而打印机只打印一行。

三、西文系统的启动打印机命令 PR#1 在汉字系统下是禁止使用的。

四、另外，"→"和"←"键也可关闭打印机。

§ 13.2 打印机控制码

（1）**字形控制**：用 POKE40963，N（16 进制 \$A003，其中 N 为代表字型的号码，是 1～8 的整数，默认值为 2，当 N 大于 8 时当成 2 处理）。

（注：默认值即汉字系统启动时自动设置的值，以下同。）

表 13-1

字型号 N	字　　形	每行最多可打的汉字或西文数 MAX
1	16×16 长方	W/16
2	16×16 正方	W/32
3	24×24 长方	W/24
4	24×24 正方	W/48
5	32×32 长方	W/32
6	32×32 正方	W/64
7	16×16 左旋 90° 长方	W/16
8	16×16 左旋 90° 正方	W/32

（注：W 为打印机倍密图形方式时每行最大点阵数，如 CP80 打印机 W＝640，FX-100 打印机 W＝1632，具体可参考打印机手册），当 W/n 为非整数时，MAX 下取整。

例 2：
```
 5 POKE 40960,5        （启动打印机）
10 FOR I=1 TO 8
15 POKE 40963, I       （设置字形）
20 PRINT I;"小蜜蜂汉字"
30 NEXT I
40 POKE 40960,0        （关闭打印机）
```

```
50 POKE 40963,2      (字型返回默认值)
RUN〈回车〉
```

(2) 行距控制

用 POKE 40962,N (16 进制 $ A002, N 为 1～85 的整数，默认值为 16) 控制行距，打印机每打印完一行，走纸 N/72 英寸。

```
例 3: 10 POKE 40960,5
      20 FOR I=3 TO 13
      25 POKE 40962, I (设行距)
      30 PRINT "小蜜蜂汉字 ABCD"
      40 NEXT
      50 POKE 40960,0
      RUN〈回车〉
```

由此例可以看出，当行距小于 7 时，相邻两行中文重叠，而西文不重叠，所以当需要打印的字全为西文时，把行距置小一点，可以节省打印篇幅。

(3) 打印宽度控制

用 POKE 40961,N (16 进制 $A001, N 不大于表 3-1 的 MAX)，用户可自行设置每行可打印的汉字数。

```
例 4: 10 POKE 40960,5
      15 FOR I=1 TO 3
      20 READ N
      30 POKE 40961,N (设置宽度)
      40 FOR J=1 TO 30:PRINT "汉"
         :NEXT J
```

```
45 PRINT
50 DATA 5 ,10,20
60 NEXT I
70 POKE 40960,0
RUN〈回车〉
```

第十四章 其它技术

§ 14.1 屏幕编辑

“小蜜蜂”汉字的屏幕编辑功能同“小蜜蜂”西文系统和 APPLE-Ⅱ 完全兼容，一切操作都是相同的，不再重复。

§ 14.2 汉字内码

在汉字系统下，西文 ASCII 码在内存中占用一个字节，而每个汉字在内存中占用 3 个字节，格式为：

字节 1　字节 2　字节 3

其中：字节 1 表示汉字标识字节

字节 2 表示国标码高字节

字节 3 表示国标码低字节

这一点是用户要注意的，尤其是在读／写文件时。

§ 14.3 监控状态

同西文系统一样，在汉字系统下进入监控也用 CALL-151〈回车〉，提示符为“*”，退出监控先打 CTRL-C，再打〈回车〉，系统又回到提示符“]”。

§ 14.4 屏幕表格线

在全屏幕文本方式下，“小蜜蜂”提供了屏幕表格线，以增强屏幕显示效果，可调用下面子程序：

```
100 SHG2:HCOLOR=3
105 FOR I=16 TO 171 STEP 17
110 HPLOT 0,I TO 279,I
120 NEXT I
140 RETURN
```

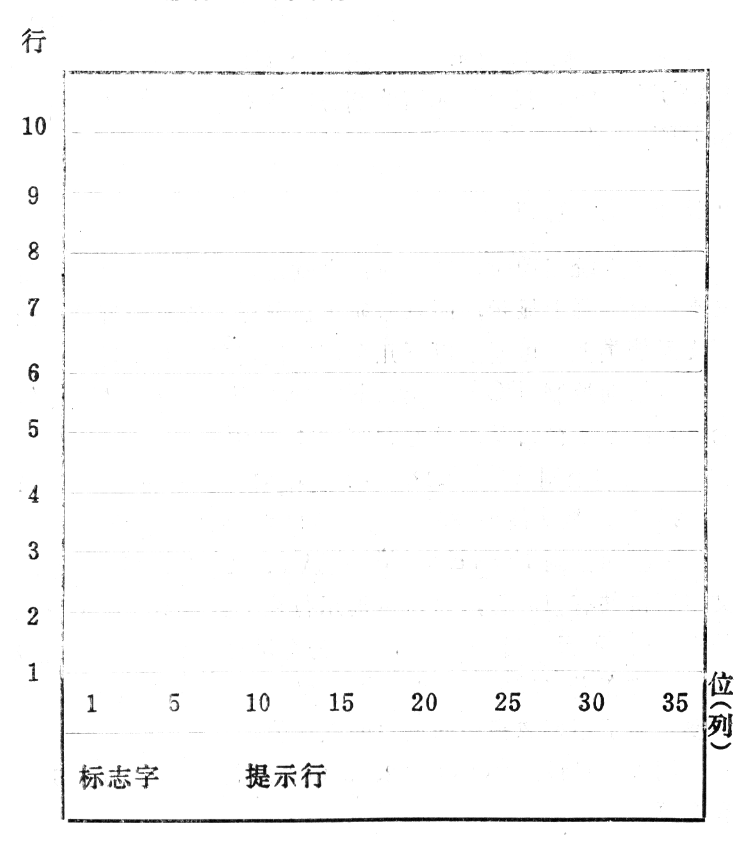
例5：

```
10 HOME:GOSUB 100
15 FOR I=1 TO 10
20 PRINT "姓名  年龄  总分"
25 PRINT "李明   18   544"
30 NEXT I
40 END
RUN〈回车〉
```

可以看到，文本行是滚动的，但表格线固定不动，除非打入清屏命令 HOME。用光标也可擦掉表格线。

第十五章　图形功能—图形与文本窗口

§ 15.1　屏幕特性与内存分配



在汉字系统下的屏幕有 10 个文本行，1个提示行，每个文本行可显示 17 个汉字或 35 个西文 ASCII 码，文本行号和列号的排列顺序如上表：

这样，APPLE-Ⅱ 的光标定位语句是：

VTAB N，N 在 1 到 10 之间

HTAB N，N 在 1 到 35 之间

“小蜜蜂”汉字系统的内存分布参见《小蜜蜂—I BASIC 使用手册》的附录 5。其中汉字系统程序在 \$A000～\$AFFF，4K。汉字显示页为高分辨率第 4 页，即 \$8000～\$9FFF，8K。

§ 15.2 文本窗口

汉字系统启动时，文本窗口也就是全屏幕。有时不需要文本在全屏幕上显示，而只需显示在屏幕上的某一部分，同西文系统类似，可通过以下几个窗口控制单元来实现。

· 横向控制 POKE 32，N1（N1 从 0～34，默认值为 0）：文本窗口从屏幕左端第 N1 号位开始。

· 横向控制 POKE33，N2（1≤N2≤35-N1 即 1≤N1+N2≤35，默认值为 35)：文本窗口为 N2 个字符宽。

· 纵向控制 POKE 34，M(M 从 14～24 默认值为 14)：文本窗口共有 10 行，窗口从提示行往上数占有 24-M 行。(不包括提示行，当 M=14 时，为全屏幕 24-14=10 行)

建议在设置横向窗口时，N1 和 N2 要同时设置，这主要是为了不要忽视横向窗口的默认值。

例 1：只设：POKE 32，5 但 33 单元的默认值 35 没有变更。

这样 N1+N2=40 超过了允许值，引起屏幕显示混乱。

例2：
```
5 HOME
10 POKE 32,5:POKE 33,10:POKE 34,20
15 HOME
18 PRINT
20 PRINT "仁仁仁仁仁仁仁仁仁仁"
RUN〈回车〉
```

请注意在第5和15行中两次清屏，第一次清的是上一次窗口，第二次清的是刚设定的窗口。

把第10行分别改为：

POKE 32, 5: POKE 33, 2: POKE 34, 15（汉字纵向显示）

POKE 32, 0: POKE 33, 35: POKE 34, 14（全屏幕文本窗口）

看看运行结果。

§ 15.3 图形窗口与绘图

当屏幕上开出一个文本窗口之后，屏幕的其余部分就叫做图形窗口，可以用 APPLE SOFT 的 HPLOT 和 DRAW 语句在图形窗口上画出图形和动画，而用文本窗口加以说明，或进行人机会话，这样通过图形和文本的交互，可以设计出很好的游戏、教学和其它软件。

例3：
```
10 POKE 32, 33: POKE 33, 2: POKE 34, 15（设文本窗口）
20 SHG2: HOME（进入图形交互方式）
```

```
30 PRINT: PRINT "蓝方块"
40 HCOLOR = 6
50 HPLOT 100,50 TO 200,50 TO 200,
  120 TO 100,120 TO 100,50
RUN〈回车〉
```

屏幕上出现一个蓝方块，并伴有汉字注释“蓝方块”。

通过以上例子，我们看到进入图形-文本交互方式要经过以下三步：

（1）设置文本窗口。

（2）用 SHG2 进入图形方式。

（3）开始绘图。

在汉字系统下的第一图形页，可供绘图的点为横向 280 点、纵向 170 点（0～169），但是由于屏幕上可同时有文本窗口和图形窗口，所以一般来讲可供绘图的点不到 280×170 点。若要避免把图画到文本窗口以内，影响屏幕效果，我们给出一个参考值：

• 全屏幕共有 10 个文本显示行，每行占纵坐标 17 个点。

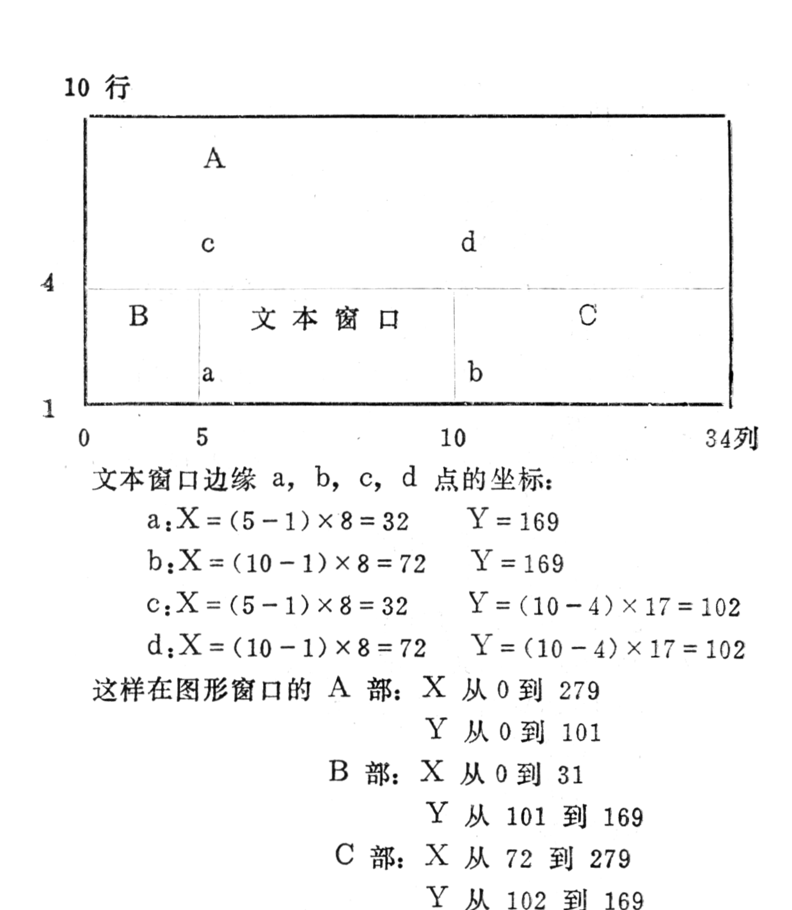
• 全屏幕每行共有 35 个字符位置，每位占横坐标 8 个点。

通过这些参考值，可以计算出屏幕可供绘图的坐标范围。

例 4：POKE 32,5：POKE 33,10：POKE 34,20

文本窗口长为 10－5＝5 位，宽为 24－20＝4 行

屏幕为：



文本窗口边缘 a，b，c，d 点的坐标：

a：$X=(5-1)\times 8=32$　　$Y=169$

b：$X=(10-1)\times 8=72$　　$Y=169$

c：$X=(5-1)\times 8=32$　　$Y=(10-4)\times 17=102$

d：$X=(10-1)\times 8=72$　　$Y=(10-4)\times 17=102$

这样在图形窗口的 A 部：X 从 0 到 279

Y 从 0 到 101

B 部：X 从 0 到 31

Y 从 101 到 169

C 部：X 从 72 到 279

Y 从 102 到 169

§ 15.4　一个实例

下面是图形-文本交互的一个实例，以供参考。

例 6：用图形矢量表画出一个图形。

看一下内存分配图，在 2048（$800）到 2560（$A00）之间有 512 字节的空白区，把图形矢量表放在 $6000开始。

打入　POKE 232，0

　　POKE 233，96

然后把图形矢量放入内存，并画图。

```
1 FOR I=0 TO 13
2 READ A
3 POKE 24576+I,A（存入图形矢量）
4 NEXT
5 DATA 1,0,4,0,18,63,32,100,45,21,54,
  30,7.0
10 POKE 34,23        （设置文本窗口）
20 SHG2
25 PRINT "旋转的图形"
30 HCOLOR=3
35 FOR I=1 TO 30
40 ROT=I:SCALE=I
50 DRAW 1 AT 139,75
70 NEXT
80 END
RUN〈回车〉
```

附　　录

附录一　XMF 汉字系统功能指标

1. 启动　在浮点 BASIC 下，用 PR#5 命令，汉字操作系统立即启动。

2. 字库

(1) 40K 压缩字库。

(2) 国标 GB2312-80 一、二级汉字。

(3) 7000 个常用词组。

(4) 简、繁体两种字体。

(5) 16*16，24*24，32*32 三种点阵。

3. 显示

(1) 显示 16*16 点阵的 GB2312-80 一、二级简繁体汉字和 8＊8 点阵的码。

(2) 10 个文本行，每行 17 个汉字或 35 个ASCII 码。

(3) 满屏汉字数：17*10＝170；
满屏西文数：35*10＝350。

(4) 一个提示行，可显示标志字和 6 个带序号的同音字或同形字。

4. 键盘

（1）支持拼音、全拼、部首、国标、区位5种汉字输入方式。

（2）支持中文、ASCII 码混合输入。

（3）支持简繁体汉字输入和 10 个表格符输入。

（4）拼音、全拼、部首输入方式支持 7000 个汉字词组输入。

（5）与 APPLE-Ⅱ 兼容的编辑功能。

（6）对误操作响铃报警。

（7）标志字注明当前键盘所处工作方式。

（8）提示行具有处理同音字和同形字的功能。

（9）乒乓键实现各种输入状态的转换。

5. 打印机

（1）打印 GB2312-80 的一、二级汉字，程序控制。

（2）打印简、繁两种字体，程序控制。

（3）打印 16*16，24*24，32*32 点阵的8种字型，程序控制。

（4）打印机可用键盘和程序控制启动/关闭。

（5）支持所有九针打印机。

（6）打印行宽可由程序控制。

（7）打印行距可在 1/72—85/72 英寸范围内由程序控制。

6. 系统功能

（1）汉字屏幕开窗口，可使图形、汉字在同屏幕上操作。

（2）可绘出 10 条不滚动的表格线。

(3) 65SC02 汇编语言可调用汉字功能。

7. 兼容性

(1) 汉字操作系统在“小蜜蜂-I”机环境下运行正常。

(2) APPLE-Ⅱ DOS3.3 命令运行无误。

(3) APPLE SOFT 命令、语句运行无误。

(4) 监控命令运行无误。

附录二 偏旁部首编码附表一、二、三

附 表 一

部首	偏旁	部首	偏旁
A	木 朩 氏 氏 氏 衣 氐 长 长 镸 民 艮	P	卩 阝 刀 力 乃 乃 卩
B	冫 氵 彡 水 氺 水 氺	Q	刂 刂 刂 目 自 且 丬 丬 爿 丬 爿 片 丬 弋 弋 戈 戈 七 气 气 戋 戊 戉
C	木 朩 木 朩	R	冂 月 用
D	八 入 𠂉 厂	S	目 耳 其 且 且 四 皿
E	丿 ㇒ 勹 夕 ⺈ 爪 夕	T	丨 十 千 午 牛 生 丰 韦 车 丰
F	一 工 五 互	U	廾 丷 𠂇
G	口 凹 凸 丘 凵 廿 艹 井 毋	V	丷 火 炏 非 灬 八 殳 癶
H	厂 广 尸 户 亻 𠂇 尸 户	W	日 田 四 臼 皿
I	凵 山 屮 匚 巨 工 彐 彐	X	土 圭 王 王 玉
J	亅 刂 了 3 ク ク 5 5 了 ㇇ マ 才 才 扌 牙	Y	丶 ㇏ 之 辶 廴 亠 言 士 业
K	豸 豸 豕 豖 马 乌 鸟 鸟	Z	卜 卜 上 止 止 此 勹 冖 宀 冎 门
L	亻 卜 乚 乙 卜 匕 七 巳 已 己 巳 巴 也		
M	小 小 ⺌ 𭕄 小 忄 少 川 巛 巛 心		
N	乂 又 攵 文 女		
O	厶 幺 幺 纟 纟 糸 乡 乡 儿 几 兀 尢 九 尤		

附表二

雨	一　冂	FR	舟	丿　⺆	ER
而	一　冂	FR	毛	丿　乚	EL
西	一　口	FG	瓜	[illegible]　丶	EY
酉	一　口	FG	鱼	⺈　一	EF
酋	丷　口	UG	矢	𠂉　人	DD
耒	一　木	FC	气	𠂉　乁	DL
身	丿　丿	EE	虫	口　丶	GY

附 表 三

A 键					
社	礻 土	A X	低	亻 氏 丶	L A Y
初	礻 刀	A P	仳	亻 长	L A
派	氵 厂 [illegible]	B H A	即	艮 卩	A P
昆	田 [illegible]	W A	眼	目 、艮	S A
表	土 丷 [illegible]	X V A	食	人 丶 艮	D Y A
展	厂 [illegible]	H A	肆	镸 ヨ 丰	A I T
农	冖 [illegible]	Z A	髻	镸 彡 士 口	A B X Y
哀	亠 口 [illegible]	Y G A	既	艮 一 尢	A F O
袁	土 [illegible]	X A			

B 键					
沿	氵 口 儿	B G O	承	氶 一 △ 一	B F F
流	氵 木	B C	函	氶 凵	B I
澡	氵 口 △ 木	B G C	流	氵 亠 △ 乚	B Y L
杉	木 彡	C B	泳	氵 丶 水	B Y B
汞	工 水	F B	淤	氵 亠 勹 ⺀	B Y P B
暴	曰 共 氺	W G B	澎	氵 士 △ 彡	B X B
冰	丶 水	Y B	溺	氵 ㇇ △ 冫	B J B
丞	氶 一	B F	习	刁 冫	J B

C 鍵

森	木 木 木	CCC	朱	𠂉 木	DC
杀	乂 木	NC	来	一 丷 木	FVC
株	木 一 朱	CFC	米	丷 木	VC
硃	厂 口 丿 朱	HGEC	粑	丷 木 巴	VCL
耙	一 木 巴	FCL	禾	丿 木	EC
枣	木 冂 ⺀	CRB	和	丿 木 口	ECG
賴	木 口 △ 人	CGD	束	木 口	CG
栗	木 罒	CW			

D 鍵

全	人 王	DX	失	𠂉 一 人	DFD
金	人 王	DX	朱	𠂉 木	DC
銀	人 王 艮	DXA	今	人 丶 フ	DYJ
飯	𠂉 ㇙ △ 又	DLN	令	人 丶 マ	DYJ
余	入 氺	DB	久	丿 人	ED
年	𠂉 丰	DT	幼	ス エ 力	DFP
午	𠂉 十	DT	算	人 人 耳	DDS
每	𠂉 母	DG	矮	𠂉 人 △ 女	DDN
舞	𠂉 一 △ 丰	DFT	氣	𠂉 乙 △ 十	DLT
乍	𠂉 ⺊	DQ			

E 键

白	丿 日	E W	乎	ソ ソ △ 亅	E V J
皇	丿 日 王	E W X	手	丿 一 △ 亅	E F J
自	丿 目	E S	先	丿 土 儿	E X O
鼻	丿 目 △ 卄	E S U	我	丿 扌 戈	E J Q
乒	丿 丘 丿	E G E	系	丿 糸	E O
乓	丿 丘 丶	E G Y	秉	丿 十 △ 人	E T D
兵	丿 丘 八	E G V	乘	丿 十 △ 人	E T D
告	丿 土 口	E X G	[illegible]	ソ 扌 人 亅	E T D J
靠	丿 土 △ 非	E X Q	拜	丿 扌 △ 丰	E T T
生	丿 土	E X	外	夕 卜	E Z
壬	丿 士	E X	受	爫 冖 又	E Z N
廷	丿 士 廴	E X Y	爱	爫 冖 𠂇 又	E Z H N
鬼	丿 田 △ 厶	E W O	爬	爪 巴	E L
重	丿 十 △ 土	E T X	抓	扌 爪	J E
垂	丿 十 △ 土	E T X	象	夕 豕	E K

F 键

大	一 人	F D	太	一 人 丶	F D Y
夫	一 一 人	F F D	未	一 木	F C
天	一 一 人	F F D	耗	一 木 丿 L	F C E L

F 键

春	一 一 一 日	FFFW	烦	火 一 △ 人	VFD
可	一 口 亅	FGJ	便	亻 一 曰 乂	LFWN
于	一 一 亅	FFJ	甫	一 冂 △ 丶	FRY
示	一 一 小	FFM	辅	车 一 冃 丶	TFRY
不	一 小	FM	专	一 一 △ 丶	FFY
否	一 小 口	FMG	转	车 一 △ 丶	TFY
丕	一 小 一	FMF	歹	一 夕	FE
亚	一 业	FY	严	一 业 丿	FYE
晋	一 业 日	FYW	丙	一 冂 人	FRD
普	丷 业 日	UYW	两	一 冂 人 人	FRDD
恶	一 业 心	FYM	下	一 卜	FZ
迈	一 勹 辶	FPY	贡	工 冂 人	FRD
逗	一 口 丷 辶	FGUY	吾	五 口	FG
豆	一 口 丷	FGU	正	一 止	FZ
吏	一 口 乂	FGN	丐	一 止 亅	FZJ
更	一 曰 乂	FWN	巫	工 人 人	FDD
丈	一 乂	FN	芈	一 丨 △ 丰	[illegible] T

G 键

国	口 王 丶	G X Y	甘	廿 一	G F
固	口 十 口	G T G	贯	口 十 △ 人	G T D
吕	口 口	G G	中	口 丨	G T
器	口 口 一 口	G G F G	面	一 丿 △ 口	F E G
共	廿 八	G V	贵	口 丨 △ 人	G T D
黄	廿 田 八	G W V	靚	井 一 冂 儿	G F R O
革	廿 口 十	G G T	喝	口 曰 勹 L	G W J L
世	廿 L	G L	跋	口 止 △ 乂	G Z N

H 键

压	厂 土 丶	H X Y	矶	厂 口 几	H G O
反	厂 又	H N	庆	广 一 人	H F D
斤	厂 一 丨	H F T	病	广 冫 △ 人	H B D
斥	厂 一 卜	H F Z	皮	广 又	H N
质	厂 十 △ 人	H T D	波	氵 广 又	B H N
灰	ナ 又	H N	颇	广 又 △ 人	H N D
发	ナ 又 丶	H N Y	虎	卜 厂 七 几	Z H L O
存	ナ 丨 △ 一	H T F	虐	卜 厂 七 ヨ	Z H L I
在	ナ 丨 土	H T X	屋	尸 一 △ 土	H F X
石	厂 口	H G	房	户 亠 方	H Y P

H 键

所	丿 コ 丿 丨	HJHT	局	尸 ㄱ 口	HJG
般	丿 ヨ △ 乂	HIN	屐	尸 土 㐅	HGA
鹿	广 コ △ 匕	HIL	層	尸 丨 目	HTS

I 键

岁	山 夕	IE	塑	丷 屮 △ 土	UIX
出	山 山	II	匈	勹 乂 凵	JNI
区	匚 乂	IN	勺	勹 丶	JY
臣	匚 丨 △ 丨	ITT	巨	匚 コ	II
聿	ヨ キ	IT	印	𠂉 卩	IP
隶	ヨ 氺	IB	兜	匚 丿 △ 儿	IEO
凶	乂 凵	NI	肅	ヨ 丨 △ 丨	ITT
幽	幺 幺 凵	OOI			

J 键

句	勹 口	JG	杨	木 ろ △ 丿	CJE
与	5 一	JF	刁	ㄱ ノ	JE
号	口 丂	GJ	买	乛 冫 △ 人	JBD
勿	勹 丿 丿	JEE	卖	十 乛 △ 人	TJD
弱	乛 丂 △ 冫	JJB	子	了 丶	JY

J 键

弓	ㄱ 丂	J J	疏	ㄱ 止 亠 ㄥ	J Z Y L
张	ㄱ 丂 长	J J A	材	木 才	C J
疆	ㄱ 丂 三 一	J J X F	村	木 寸	C J
剁	ノ 木 刂	E C J	扛	扌 工	J F
甬	ㄱ 用	J R	牙	一 牙	F J
予	ㄱ 一 亅	J J J	呀	口 一 牙	G F J
矛	ㄱ 一 ㄧ 亅	J J E	丑	ㄱ 𠂉 一	J T F
子	了 一	J F	司	亅 一 口	J F G
孑	了 ノ	J E			

K 键

狼	犭 丶 良	K Y A	鸭	日 丨 鸟	W T K
貂	豸 刀 口	K P G	凫	鸟 几	K O
象	⺈ 罒 豕	E W K	逐	豕 辶	K Y
家	宀 豕	Z K	遂	丷 豕 辶	U K Y
驼	马 宀 匕	K Z L	啄	口 豕 丶	G K Y
鸵	鸟 宀 匕	K Z L	狍	犭 勹 巳	K J L
吗	口 马	G K	燹	豕 豕 火	、K Y

L 鍵

们	亻 门	L Z	把	扌 巴	J L
很	亻 亻 艮	L L A	地	土 也	X L
行	亻 亻 一 亅	L L F J	孔	了 一 乚	J E L
何	亻 一 口 亅	L F G J	飞	乁 ㄑ	L B
民	己 七	L Q	鼠	臼 ㇀ △ ㇃	W L L
改	己 攵	L N	曷	曰 勹 乚	W J L
异	巳 廾	L U	甚	其 乚	S L
仓	人 已	D L	兆	丬 乚	Q L
仑	人 匕	D L	北	丬 匕	Q L

M 鍵

尖	小 一 人	M F D	恭	廾 八 小	G V M
尚	小 冂 口	M R G	少	小 丿	M E
党	⺌ 冖 口 儿	M Z G O	省	小 丿 目	M E S
学	⺍ 冖 △ 一	M Z F	步	止 少	Z M
敞	⺌ 冂 △ 攵	M R N	州	丷 川 丶	M M
敝	丷 冂 △ 攵	M R N	惟	忄 亻 △ 土	M L X
应	广 ⺍ 一	H M F	志	士 心	X M
佥	人 一 △ 一	D F F	巢	巛 田 木	M W C
光	小 兀	M O	粼	丷 木 △ 巛	V C M

N 鍵

圣	又 土	N X	夜	亠 亻 厶 夂	Y L N
义	丶 乂	Y N	杀	乂 木	N C
邓	又 阝	N P	希	乂 𠂇 厶 丨	N H T
故	十 口 攵	T G N	逢	夂 丰 辶	N T Y
齐	文 川	N Q	风	几 乂	O N
如	女 口	N G	离	文 凵 丁 丶	N I R Y
处	夂 卜 丶	N Z	螽	夂 丶 厶 丶	N B Y

O 鍵

允	厶 儿	O O	幻	幺 丁	O J
充	亠 厶 儿	Y O O	红	纟 工	O F
育	亠 厶 月	Y O R	龙	尤 丿	O F
云	一 一 厶	F F O	机	木 几	C O
魂	一 一 厶 厶	F F O O	系	丿 糸	E O
矣	厶 𠂉 人	O D D	尧	戈 兀	Q O
卯	𠂎 卩	O P	旭	九 日	O W
卿	𠂎 目 卩	O A P	兄	口 儿	G O
卵	𠂎 卩 丶	O P Y			

P 键

扔	扌 乃	J P	忍	刀 丶 心	P Y M
扳	扌 尸 又	J P N	阵	阝 车	P T
极	木 乃 乀	C P Y	陈	阝 一 木	P F C
及	乃 乀	P Y	办	力 八	P V
召	刀 口	P G	尕	乃 小	P M
郎	亠 彐 △ 阝	Y U P	勰	力 力 △ 心	P P M
另	口 力	G P	隧	阝 彐 △ 辶	P U Y
盈	乃 又 皿	P N C			

Q 键

坚	刂 又 土	Q N X	殷	片 丫 又	Q H N
帅	刂 冂 丨	Q R T	式	弋 工	Q F
阜	𠂤 十	Q T	成	戊 丁	Q J
官	宀 㠯	Z Q	武	弋 止	Q Z
假	亻 𠃑 △ 又	L Q N	栽	𢦏 木	Q C
锻	钅 几 又	Q O N	钱	人 钅 戋	D X Q
[illegible]	丬 𠂤	Q Q	越	土 龰 戉	X Z Q
收	丩 攵	Q N	盥	乡 水 △ 皿	Q B S
壮	丬 士	Q X	舋	⺈ 冂 △ 丶	Q R Y
臧	丬 戈 △ 丨	Q Q T			

R 键

鹏	月 月 鸟	RRK	帐	冂 丨 长	RTA
同	冂 一 口	RFG	丹	𠃌 亠	RY
角	⺈ 用	ER	扁	户 冂 卄	HRU
解	⺈ 用 刀 牛	ERPT	需	一 冖 △ 冂	FRR
甩	用 乚	RL	霄	一 冖 ⺌ 月	FRMR
网	冂 ㄨ ㄨ	RNN	页	一 丿 冂 人	FERD
册	𠃌 𠃌 一	RRF			

S 键

眼	目 艮	SA	真	十 且 八	TSV
取	耳 又	SN	罹	罒 巾 △ 土	SMX
敢	工 耳 攵	FSN	盅	口 丨 皿	GTS
其	甘 八	SV	血	丿 皿	ES
蛆	口 丶 且	GYS	斯	甘 八 ⺁ 丨	SVHT

T 键

古	十 口	TG	伟	亻 韦	LT
犁	丿 木 刂 牛	ECJT	喜	車 口	TG
牯	牛 十 口	TTG	降	阝 夂 㐄	RNT
轩	车 一 十	TFT	攸	亻 丨 攵	LTN

T 键

寿	𡗗 寸	T J	鬲	一 口 冂 丨	F G R T
旧	丨 日	T W	市	亠 冂 丨	Y R T
齿	丰 凵	T I			

U 键

苦	廿 十 口	U T G	辛	亠 丷 十	Y U T
前	丷 月 刂	U R J	幸	土 丷 十	X U T
黄	丷 口 一 人	U G F D	算	⺮ ⺮ 目 廾	U U S U
半	丷 十	U T	奔	一 人 △ 廾	F D U
叛	丷 丰 ㄏ 又	U T H N	卷	丷 一 人 㔾	U F D L
羊	丷 丰	U T	酋	丷 丿 目	U E S
立	亠 丷	Y U	竺	⺮ ⺮ △ 一	U U F
豆	一 口 丷	F G U	扁	户 冂 廿	H R U
南	𠂇 冂 △ 十	T R T			

V 键

公	八 厶	V O	父	八 乂	V N
分	八 刀	V P	弟	丷 フ △ 丿	V J E
灶	火 土	V X	赭	土 小 土 日	X V X W
照	日 刀 △ 灬	W P V	办	力 八	P V

V 鍵

亦	亠 小	Y V	脊	癶 月	V R
奕	亠 小 △ 人	Y V D	登	癶 一 △ 丷	V F U
叢	丷 彐 小	U I V	祭	癶 一 △ 小	V F M
赤	土 小	X V			

W 鍵

明	日 月	W R	典	曰 丨 △ 八	W T V
略	田 夂 口	W N G	曹	廾 曰 日	U W W
曾	丷 四 日	V W W	昌	日 曰	W W
黑	四 土 灬	W X V	晶	日 日 日	W W W
點	四 土 灬 口	W X V G	舅	臼 田 力	W W P
曲	曰 丨 丨	W T T			

X 鍵

吉	士 口	X G	集	亻 亠 △ 木	L Y X C
青	主 月	X R	堆	土 亻 △ 土	X L X
地	土 也	X L	坞	土 马	X K
主	丶 王	Y X	琶	王 王 巴	X X L
佳	亻 土 土	L X X	冉	冂 土	R X
住	亠 土	L Y X	再	王 冂	X R

Y 鍵

六	亠 八	YV	哀	亠 口 𧘇	YGA
商	亠 丷 △ 口	YVG	玄	亠 幺	YO
立	亠 䒑	YU	信	亻 言	LY
端	亠 䒑 △ 冊	YUR	讲	讠 井	YG
竟	亠 䒑 △ 儿	YUO	刃	刀 丶	PY
产	亠 䒑 丿	YUE	羸	亠 乚 △ 丶	YLY
彦	亠 䒑 △ 彡	YUB	盲	亠 乚 目	YLS

Z 鍵

卢	⺊ 尸	ZH	赵	土 龰 乂	XZN
外	夕 卜	EZ	走	土 龰	XZ
虎	⺊ 一 七 几	ZHLO	欠	𠂊 人	ZD
卡	上 卜	ZZ	尔	𠂊 小	ZM
此	止 匕	ZL	你	亻 𠂊 小	LZM
正	一 止	FZ	们	亻 门	LZ
步	止 少	ZM	骨	冎 月	ZR
跟	口 止 艮	GZA	髋	冎 月 △ 儿	ZRO

下　篇

中华学习机“小蜜蜂-I”(XMF-I)型微型计算机技术参考手册

第一章　系统结构与中央处理器

§ 1.1　系统结构

中华学习机“小蜜蜂-I”微型计算机是一个 8 位的个人计算机，它是由主控板、主机箱、键盘集装于一体，形成一个便携式的主机装置。结构设计中采用了积木式结构特点，在主机装置上，留有电源插口、录音机输入、输出插口、监视器插口、彩色或黑白电视机插口、游戏棒插口、打印机插口以及带有软磁盘驱动器的扩展箱插口。将主机与必要的显示设备、外存储设备、打印机设备连接，就可形成各种不同配置的微型计算机系统。原理图如图 1-1 所示。

主控板是由双面附铜板制作的，尺寸为：340mm×298mm。主控板采用总线型结构，地址线 16 条，数据线 8 条，控制线 5 条。中央处理器为 65SC02，与时序电路、视频电路、存储器及地址译码电路、输入/输出接口及扩展槽电路、汉字系统电路等组成主控板运行工作电路。

§ 1.2　中央处理器 65SC02（CPU）

65SC02 CPU 是用 CMOS 半导体工艺生产的大规模集成电路（LSI），它是一种典型的总线结构形式，具备功耗低、速度快等特点。单电源 +5V，地址总线 16 条，数据总线 8 条，控制总线 7 条，时钟信号线 3 条，电源及地线

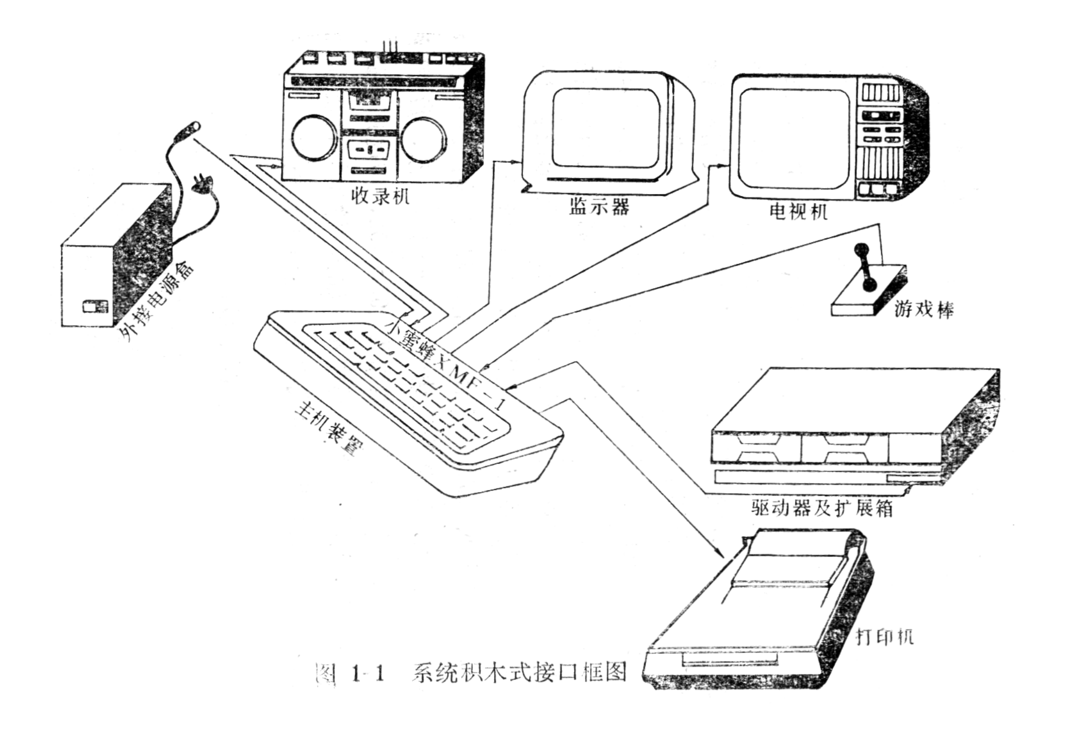


图 1-1　系统积木式接口框图

共占了 3 条，尚有三条未使用的引线，总共 40 条引腿，包封在工业标准 40 条双列直插式陶瓷或塑料管壳中。

65SC02 CPU 的主频为 1MHz，时钟周期为 1μs，直接寻址能力 64K，指令 64 条，15 种寻址方式，形成 178 个操作码，与 6502 CPU 完全兼容，其中指令系统包含了 6502 CPU 的全部 56 条指令。它的管腿封装定义如图 1-2 所示。

65SC02 信号之间的时序关系如图 1-3 所示。

65SC02 CPU 时序图中的时间参数关系如表 1-1 所示。

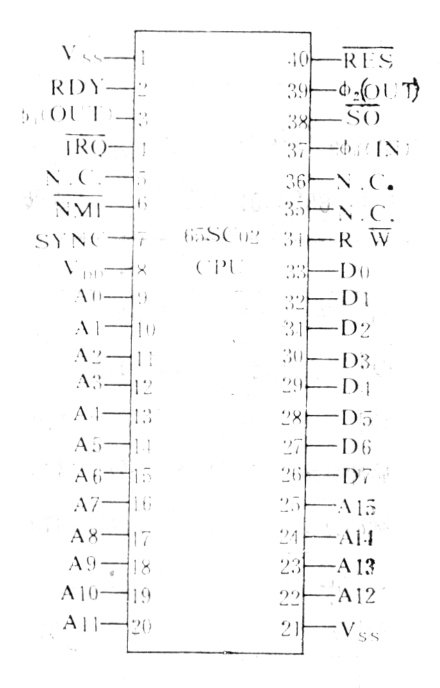


图 1-2 65SC02 引腿定义

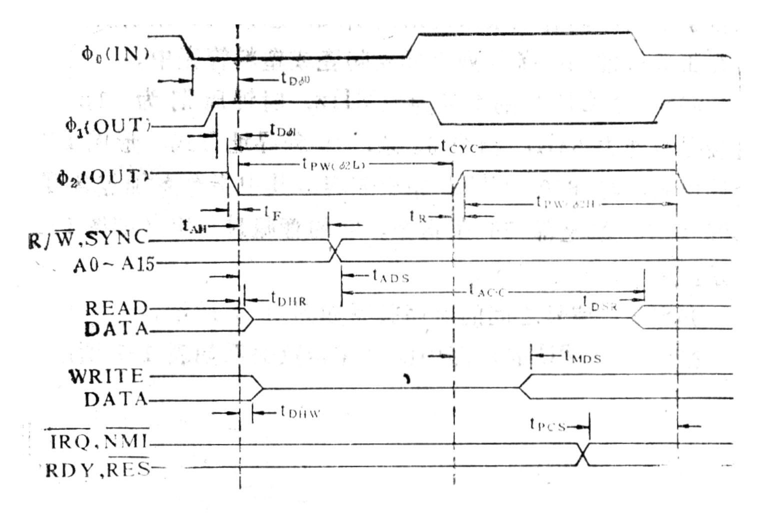


图 1-3 65SC02 CPU 时序图

表 1-1 65SC02 CPU 时间参数

定　　义	符　合	最小	最大	单位
Φ0(IN)到Φ2(OUT)的延迟时间	TDΦ0		100	ns
Φ1(OUT)到Φ2(OUT)的延迟时间	TDΦ1		50	ns
周期时间	TCYC	1.0	DC	μs
时钟负半周脉冲宽度	TPW(Φ2L)	470	—	ns
时钟正半周脉冲宽度	TPW(Φ2H)	470	—	ns
下降沿时间，上升沿时间	TF,TR	—	25	ns
地址保持时间	TAH	30	—	ns
地址建立时间	TADS	—	225	ns

续表

地址有效时间	TACC	650	—	ns
读数据保持时间	TDHR	10	—	ns
读数据建立时间	TDSR	100	—	ns
写数据保持时间	TDHW	30	—	ns
写数据延迟时间	TMDS	—	175	ns
处理器控制信号建立时间	TPCS	200	—	ns

65SCO02 CPU 指令系统详见书末指令表。

§1.3 系统总线

XMF-I 采用总线型系统结构。地址总线 16 条，数据总线 8 条，控制总线 7 条。地址码经地址总线驱动器74LS244（三态控制门）送出地址码，数据信息则经过双向数据总线驱动器（74LS245 三态控制门）将指令码、状态或数据信息送出或读入。

在 7 条控制总线中，输入线有 5 条，它们是：

$\overline{\text{IRQ}}$ 可屏蔽中断请求。

$\overline{\text{NMI}}$ 非屏蔽中断请求。

RDY 准备好信号。当为高电平时，表示存储器读准备好；当为低电平时，表示存储器读速度慢，请求 65SC02 读周期延迟。

$\overline{\text{RES}}$ 复位信号。

$\overline{\text{SO}}$ 用于使溢出标志置“1”（系统中未用，接低电平）的信号。

输出的控制线有 2 条：

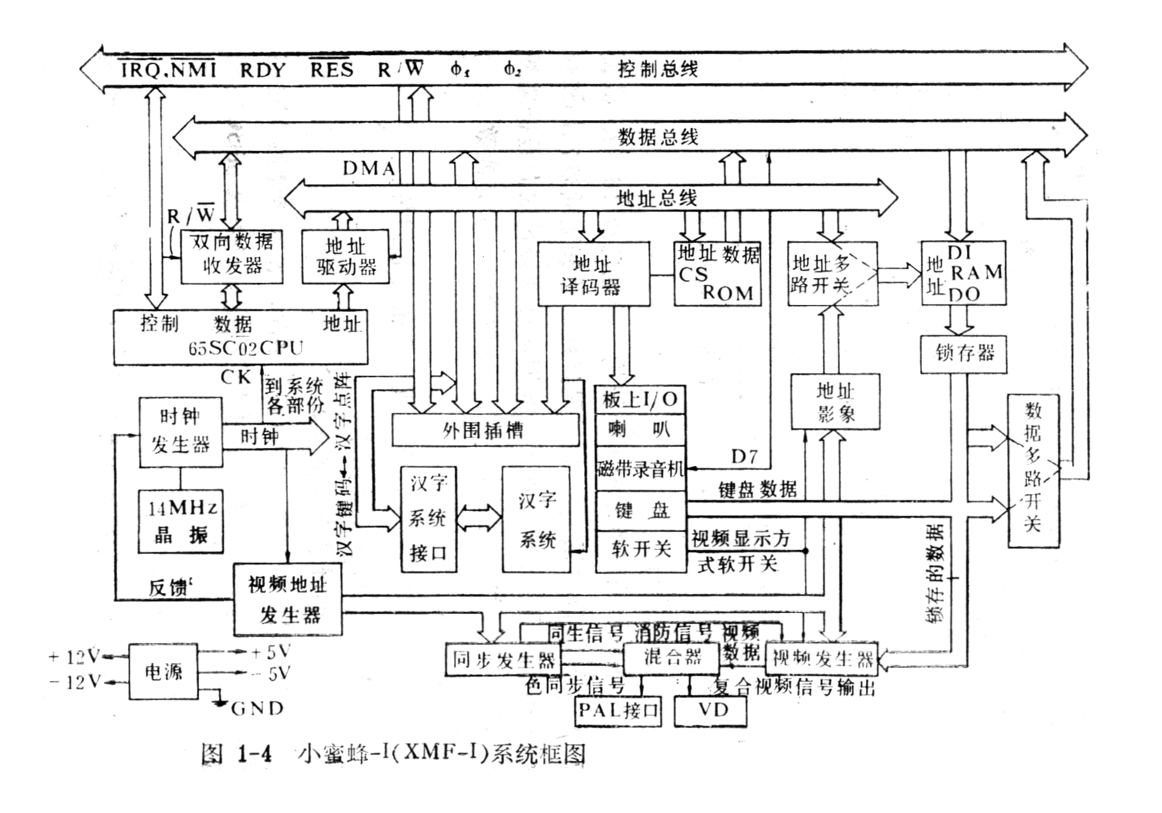


图 1-4 小蜜蜂-I(XMF-I)系统框图

SYNC　读操作码时为高电平。

R/$\overline{W}$　读/写信号。

系统的时钟信号为：

Φ0　输入时钟，频率为 1MHz。

Φ1　CPU 输出时钟，频率为 1MHz，与 Φ0 反相。

Φ2　CPU 输出时钟，频率为 1MHz，与 Φ0 同相。

系统的结构框图如图 1-4 所示。

§ 1.4　系统内存地址分配

XMF-I 内存空间直接寻址能力为 64K 字节（今后如无特殊说明，均以字节为单位计算内存容量），存储器访问与 I/O 口地址统一编址。其空间分配如图 1-5 所示：

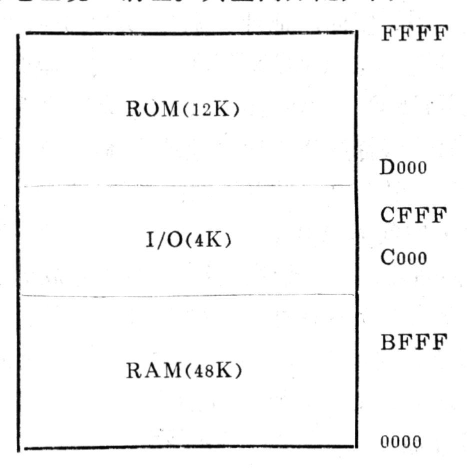


图 1-5　内存空间分配示意图

三个地址区分别为：

RAM 48K \$0000～\$BFFF

I/O　4K　$C000～$CFFF

ROM　12K　$D000～$FFFF

在地址空间中，定义 256 个字节为一个地址页。

RAM 区的 48K 地址空间，不能全部分配给用户使用，有些被确定为专用的区域，如系统的零页区、堆栈区、键盘输入数据缓冲区、系统向量区、汉字处理程序区等。由于系统未设专门的显示刷新存储区，因此在 RAM 存储区中专门开辟了字符、图形页，所以 RAM 地址应该允许 CPU 和 CRT 分时进行访问。

输入/输出（I/O）口地址分配在 $C000～$CFFF 4K 空间，作为 I/O 口地址、扩充槽口地址、系统软开关、汉字、软磁盘驱动器及打印机等引导程序地址。

ROM 区 12K 空间，地址分配在 $D000～$FFFF，其中分配给监控程序区（MONITOR）使用最高位的 2K地址 $F800～$FFFF，另外 10K 空间作为固化程序语言区，存放 BASIC 语言解释程序，地址为 $D000～$F7FF。但是在 XMF-1 型机中，采用了增强型的 XMF-BASIC 语言，要求固化空间为 14K，因此采用了 32K EPROM (27256) 只读存储器，其中最高 2K 字节仍存放监控程序，余下的 30K 分为三个存储体，分别命名为 0 号、1 号、2 号存储体，每个存储体都为 10K 空间。三个体之间的切换用软开关进行。其绝对地址空间分别为：

0 号存储体：$D000～$F7FF

1 号存储体：$A800～$CFFF

2 号存储体：$8000～$A7FF

在选择存储体时，0 号体的绝对地址是系统分配的有效

地址，因此只需将体号与 0 体的绝对地址影象到相应体号对应的绝对地址，即可达到转体的目的。

存储器的地址分配如图 1-6 所示。

存储区			地址
			FFFF
监 控 程 序			
			F800
0 体 ROM 10 K	1 体 ROM 10 K	2 体 ROM 10 K	
			D000
I/O 口地址（4K）			
			C000
汉字处理程序（4K）			
			A000
第四图形页（8K）汉字字符点阵缓冲区			
			8000
第三图形页（8K）			
			6000
第二图形页（8K）			
			4000
第一图形页（8K）			
			2000
用 户 区（5K）			
			0C00
第二字符图形页（1K）			
			0800
第一字符图形页（1K）			
			0400
系统向量区（256）			

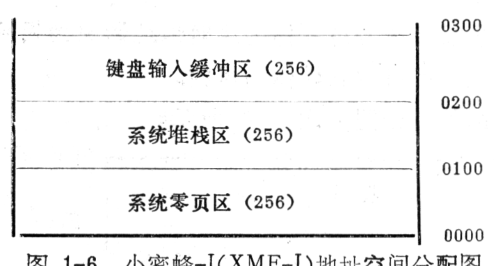


图 1-6　小蜜蜂-I(XMF-I)地址空间分配图

由于 65SC02 CPU 只有 16 位地址线，因此限定了直接寻址空间只有 64K，但该系统规定，其中高 12K（$FFFF～$D000）为系统 ROM 区（存放监控程序及 BASIC 语言解释程序），接下来 4K（$CFFF～$C000）是分配给 I/O 口地址使用的地区，以下为 48K RAM 区。将系统内存扩充为 64K，必须增加 16K 空间，其根据是：

1. 系统调入 DOS 3.3 操作系统后，ROM 中提供的

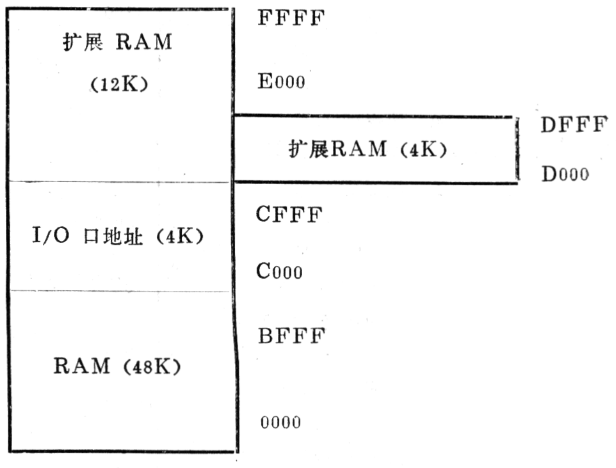


图 1-7　16K RAM 扩展地址分配区

固化监控程序和 BASIC 语言解释程序可以不用，因此让出来 12K 空间。

2. 此时 I/O 口地址 4K 空间绝对不能占用。

3. 利用空出来的 12K 空间，并用体选低 4K 地址重叠的技术，可以达到扩充 16K RAM 空间的目的。

其扩展 16K RAM 空间图如图 1-7 所示。

为了扩充用户使用的随机存储区，XMF 系统采用了 DOS3.3 搬家的技术。当由软磁盘将 DOS3.3 调入内存后，存放在 $BFFF～$9600 区域，这正是 48K 区域的一块空间。为了扩充用户使用的连续空间，在进入汉字系统或在 XMF-BASIC 状态下，用 CALL-$77A 命令都可使 DOS3.3 转移到 $FFFF 开始的以下空间内。

§ 1.5 输入、输出（I/O）口地址

I/O 口地址空间为4K，但实际并未全部作为外设口地址。其中定义了某些板上固定的口地址，还定义了扩充槽口的选通地址，有些则用来作为软开关及可编程的智能地址。

定义为板上固定 I/O 口地址的共8个($C000～$C070)，它们是：

$C070	游戏机摇杆触发器。
$C060～$C06F	盒带及游戏选通信号。
$C050～$C05F	显示工作方式及游戏开关量。
$C040～$C04F	扩充软开关及游戏实用选通信号。
$C030	触发扬声器。
$C020	触发盒带输出。
$C010	清键盘选通。

$C000　键盘数据输入。

$C040～$C04F 用来作为 XMF-I 的扩充软开关使用，定义如下：

$C04F　开　选通第三、四图形页。

$C04E　关　选通第一、二图形页。

$C04D　开　选 ROM 0 存储体。

$C04C　关　选 ROM 1 存储体。

$C04B　开　选 ROM 0 存储体。

$C04A　关　选 ROM 2 存储体。

$C045　开　汉字单色状态。

$C044　关　汉字彩色方式。

$C042　关　发声。

$C040　关　游戏实用选通。

$C050～$C05F 软开关定义如下：

$C05F　开　AN3 主机通过游戏口输出指示信号。

$C05E　关　AN3 同上。

$C05D　开　AN2 同上。

$C05C　关　AN2 同上。

$C05B　开　AN1 同上。

$C05A　关　AN1 同上。

$C059　开　AN0 同上。

$C058　关　AN0 同上。

$C057　开　HIRES MODE　选择高分辨率。

$C056　关　HIRES　选择低分辨率。

$C055　开　PAGE2　选择第二页。

$C054　关　PAGE2　选择第一页。

$C053　开　MIX MODE 选择混合方式。

$C052　关　$\overline{\text{MIX MODE}}$ 选择全文本、图形方式。

$C051　开　TEXT MODE 选择全文本方式。

$C050　关　$\overline{\text{TEXT MODE}}$ 选择图形方式。

$C060～$C06F 用来产生游戏机输入主机三个开关信号，四个模拟信号及收录机转储（输入）接口软开关量。

$C067 或 $C06F　游戏输入模拟信号 PDL-3。

$C066 或 $C06E　游戏输入模拟信号 PDL-2。

$C065 或 $C06D　游戏输入模拟信号 PDL-1。

$C064 或 $C06C　游戏输入模拟信号 PDL-0。

$C063 或 $C06B　游戏输入开关量 SW2。

$C062 或 $C06A　游戏输入开关量 SW1。

$C061 或 $C069　游戏输入开关量 SW0。

$C060 或 $C068　收录机转储（输入）接口。

XMF-1 的 8 个扩展槽，其中有 4 个已占用，即 0 号槽用作开发 16K RAM，使系统的 RAM 空间可扩充到 64K；1 号槽接打印机；5 号槽接汉字；6 号槽接软磁盘驱动器；留给用户使用的尚有 2 、3 、4 、7 四个槽口。扩展槽口地址信号一般表达式为：$\overline{\text{I/O SEL}}$ CnXX。

C0XX　0 号　16K RAM 扩展专用。

C1XX　1 号　打印机接口。

C2XX　2 号　用户开发用。

C3XX　3 号　用户开发用。

C4XX　4 号　用户开发用。

C5XX　5 号　汉字接口地址。

C6XX　6号　软磁盘驱动器口地址。

C7XX　7号　用户开发用。

同时每个槽口还设有相应的 16 个可使用的地址单元，与槽口号一一对应。用户可用它们作为每个扩展槽口的专用开发地址，其选通信号的一般表达式是 $\overline{\text{I/O DEV C0nX}}$。

$C080～$C08F　对应 0 号槽。

$C090～$C09F　对应 1 号槽。

$C0A0～$C0AF　对应 2 号槽。

$C0B0～$C0BF　对应 3 号槽。

$C0C0～$C0CF　对应 4 号槽。

$C0D0～$C0DF　对应 5 号槽。

$C0E0～$C0EF　对应 6 号槽。

$C0F0～$C0FF　对应 7 号槽。

从 $C800 起到 $CFFF 这 2K 空间作为各个可扩充槽口的公用地址，用户可用作智能程序的开发空间。本系统就是利用该空间，用一片 2764 8K EPROM 作为打印机引导程序、打印机处理程序、软磁盘驱动器引导程序、汉字系统引导程序及汉字处理程序的存放区。其逻辑地址分配图如图 1-8 所示。

§ 1.6 系统性能

1. 中央处理器 CPU 65SC02。
2. 主存储器 RAM 64K 字节。
3. 只读存储器 42K 字节。
4. I/O 可寻址空间 4K 字节。
5. 外设接口

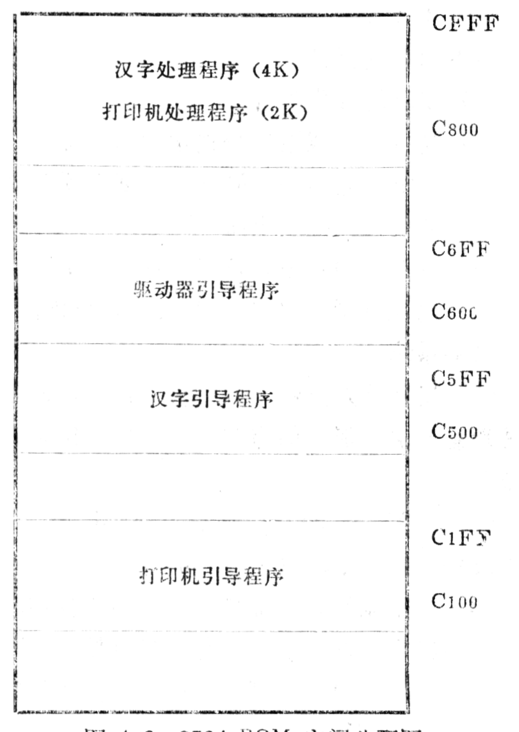


图 1-8　2764 ROM 空间分配图

（1）字符显示终端 VD 接口。

（2）电视机（彩色 PAL 制式）接口 T.V。

（3）磁带收录机接口。

（4）打印机接口。

（5）游戏机接口。

（6）喇叭接口。

（7）发声接口。

（8）扩展箱接口，包括：软磁盘驱动器，扩展插槽

4个。

6. 专用门阵列器件

(1) 视频地址发生器 GA_1。

(2) 视频、总线数据分选电路 GA_2。

(3) 随机存储器RAM行、列地址分选电路GA_4。

(4) 视频方式控制电路 GA_5、

(5) 汉字、图形方式选择电路 GA_7。

(6) 电视（彩色 PAL 制式）接口电路GA_8。

7. 汉字系统

(1) 汉字输入方法：声韵输入法，简易拼音输入法，全拼音输入法，形体偏旁部首输入法，国标码输入法，区位码输入法。

(2) 汉字库。ROM 容量 96K 字节，存放国标一、二级汉字 6763 个，词组 2500～7000个。

(3) 汉字显示。一屏幕 17×10 个汉字，可显示国标简体或繁体汉字字形。最末一行为汉字提示行。

(4) 汉字打印。可打印国标简、繁体汉字 16×16，24×24，32×32 字形。

8. 屏幕显示

a.字符显示

字符点阵 7 × 8

一帧 24×40

b.图形显示

低分辨率2个页面，每个页面 1K 空间。

象素 4 × 2 × 7

一帧 24× 2 ×40

彩色　16 种

高分辨率 4 个页面，每个页面 8K 空间。

象素　位点

一帧　192×280

彩色　6 种

9．键盘

（1）弹簧键 53 个

（2）拼音复合键

汉字拼音声母、韵母、调号及汉字形体偏旁部首码与英文字母组成 26 个复合键。键盘如图 1-9 所示。

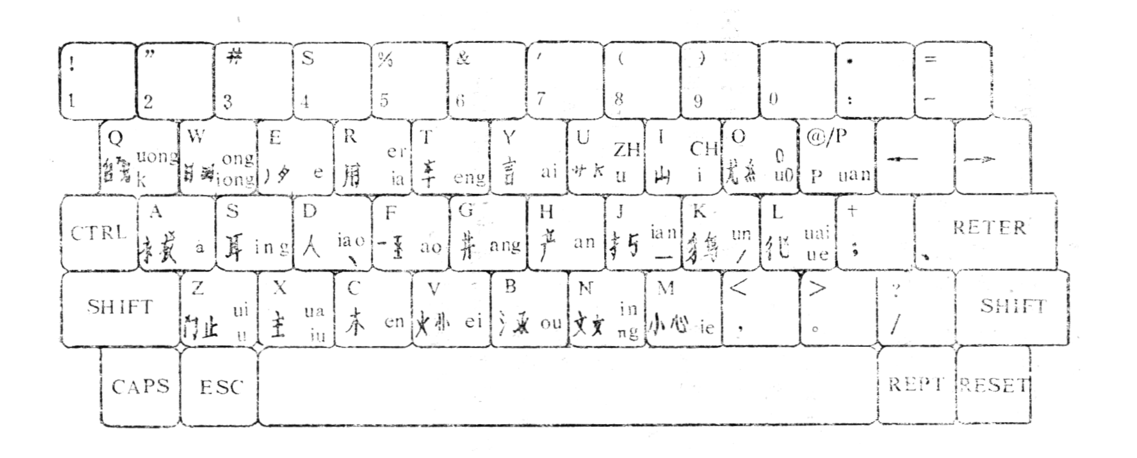
10．电源

（1）外接交流电源 220V，50Hz。

（2）外接直流 +5V 稳压，1.5A。

（3）扩展箱电源

提供软磁盘驱动器系统与主机所需 +5V，-5V，+12V，-12V 直流电源。



Q W E R T Y U I O @/P
A S D F G H J K L
Z X C V B N M
CTRL
SHIFT
CAPS
ESC
RETER
SHIFT
REPT
RESET

图 1-9 键盘码布位图

第二章　专用门阵列电路

门阵列设计技术是面向用户所需的逻辑电路进行专门设计的一种技术手段。在设计结束之后，通过半导体大规模集成电路（LSI）制作工艺，作成专用的电路芯片。因此，这种用户专用的集成电路保密性强、集成度高、运行可靠、功耗小、成本低，是提高系统性能价格比的强有力的手段之一。

"小蜜蜂-I"微型计算机系统设计时采用了门阵列设计的技术措施，为本系统设计了六个专用的门阵列器件，要求采用 CMOS 工艺制作，这样制作的芯片只相当同样功能 TTL 工艺芯片功耗的十分之一，门的延迟小于 2 毫微秒，主频大于7MHz，考虑到我国半导体制做工艺水平，门的集成度设计在 300 门到 500 门中规模范围之内，以便有可能在近两、三年内实现国产化，芯片内门的利用率达到 75% 以上，包封在工业标准 40 条引腿的塑料或陶瓷管壳中，环境条件要求温度为 0℃～70℃，湿度小于百分之九十时正常工作。六种门阵列器件的名称是：

1. GA_1　视频地址发生器
2. GA_2　视频，总线数据分选电路
3. GA_3　随机动态存储器 RAM 行、列地址发生电路

4. GA4 视频显示方式控制电路

5. GA7 汉字图形方式选择电路

6. GA8 电视（彩色 PAL制）接口

下面将对各个专用门阵列电路的引腿定义做如下介绍。

§ 2.1 视频地址发生器 GA1

GA1 门阵芯片采用了半导体 CMOS 工艺制做而成，主要用来产生视频显示有关的地址计数信号。该器件要求单一+5V 电源，包封在工业标准 40 引腿双列直插式的管壳

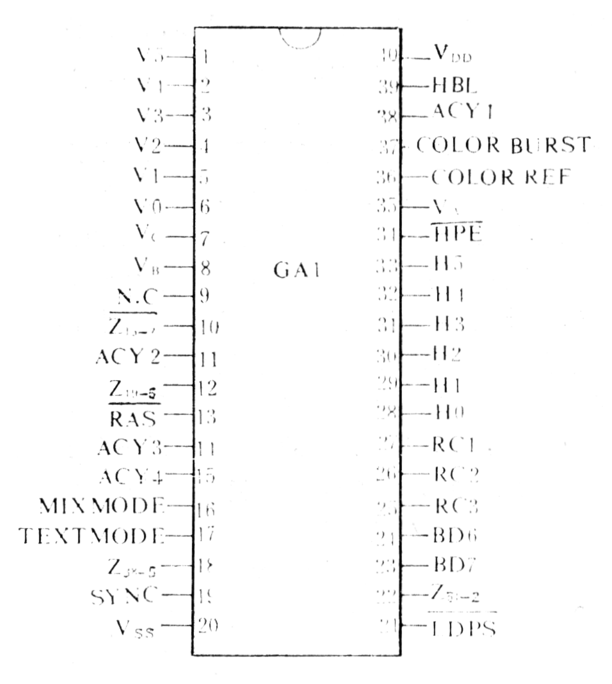


图 2-1 GA1 引腿定义图

中。其引腿定义如图 2-1 所示。

H_0, H_1, H_2, H_3, H_4, H_5, $\overline{HPE}$ ——视频显示水平时序计数信号。输出状态，引腿号分别为 28、29、30、31、32、33、34。

VA, VB, VC, V_0, V_1, V_2, V_3, V_4, V_5 ——视频显示垂直时序计数信号。输出状态，引腿号分别为 35、8、7、6、5、4、3、2、1。

ACY_1, ACY_2, ACY_3, ACY_4——跳接线连接出头，用以适应工业电频为 50Hz 或 60Hz 的工作环境。引腿分别为 38、11、14、1。

$\overline{Z_{15}\text{-}2}$ ——直接连接到 GA_5-25 的控制信号，输出状态，低电平有效。引腿为 10。

Z_{49}-5——直接连接到 IC 器件 Z_{49}-5 引腿上的控制信号，输出状态。引腿为 12。

$\overline{RAS}$——为动态随即存储器行地址选通的控制信号。低电平有效。由 IC 器件 Z_{42}-1 腿送来。输入状态。引腿号为 13。

MIXMODE——字符与图形混合方式选通信号，由 GA_5-31 送来，输入状态，高电平状态为文字与图形混合显示方式。引腿号为 10。

TEXTMODE——文本图形方式选通信号，由GA_5-31 送来，输入状态，高电平状态为文本显示方式。引腿号为17。

Z_{38}-5——直接连接到 IC 器件 Z_{38}-5 引腿上的控制信号。输出状态。引腿号为 18。

SYNC——视频电路控制信号，直接连接到 Z_{58a}-5，4 引腿上。输出状态。引腿号为 19。

$\overline{LDPS}$——视频电路水平及垂直计数的触发脉冲信号。输入状态。直接连接到 Z113-8。引腿号为 21。

Z51-2——直接连接到 IC 器件 Z51-2 和 Z38-6 上的控制信号。输出状态。引腿号为 22。

BD7,BD6——由 GA_2 送来的视频数据信息，输入状态。引腿号分别为 23，24。

RC_3，RC_2，RC_1——与电容 C101 和电阻 R112 相连接的连接信号。引腿号分别为 25，26，27。

COLOR REF——NTSC 制3.5MHz 的彩色基准信号。输入状态。引腿号为 36。

COLOR BURST——产生色同步脉冲的控制信号，输出状态。引腿号为 37。

HBL——行消隐信号，输出状态。引腿号为 39。

VSS——为 GA_1 器件的“源”极接地。引腿号为20。

VDD——为 GA_1 器件的“漏”极接＋5V。引腿号为40。

§2.2 视频与总线数据分选电路 GA_2

GA_2 门阵列芯片采用了半导体 CMOS 工艺制做而成，用来把动态存储器 RAM（4164）输出的数据，或从键盘输入的数据，分别送往视频电路或数据总线。该器件要求单一＋5V 电源，包封在工业标准 40 条引腿双列直插式的管壳中。其引腿定义如图 2-2 所示。

AX——分选电路控制信号。从 IC 器件 Z82-9 直接送出连接到 GA_2-1。输入状态。引腿号为 1。

DO_0，DO_1，DO_2，DO_3，DO_4，DO_5，DO_6，DO_7——动态存储器 RAM（4164）输出端送出的数据信号。输出状

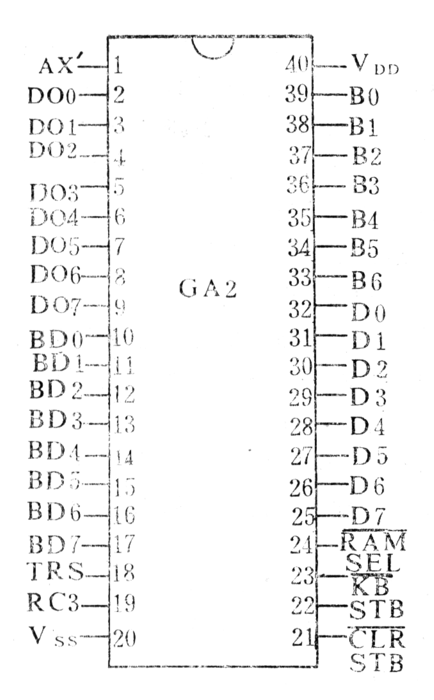


图 2-2 GA2 引腿定义图

态。引腿号分别为 2，3，4，5，6，7，8，9。

BD0，BD1，BD2，BD3，BD4，BD5，BD6，BD7——视频及字符发生器所需数据信号。输出状态。引腿号分别为 10，11，12，13，14，15，16，17。

RC3——计数器所需外接 RC 电路连接信号。引腿号为 19。

TRS——电源接通（上电）产生$\overline{\text{RESET}}$时的控制信号。输出状态。直接连接到 R115 电阻上。引腿号为 18。

VSS——为 GA_2 器件的“源”极接地。引腿号为20。

$\overline{\text{CLRSTB}}$ ——清键盘选通的控制信号。与 GA_5-10连接。输入状态。低电平有效。引腿号为 21。

STB——键盘选通信号。与键盘插座 KBSLOT-5 相连。输入状态。引腿号为 22。

$\overline{\text{KB}}$ ——键盘数据输入允许信号。输入状态，低电平有效。与 GA_5-9 相接。引腿号为 23。

$\overline{\text{RAMSEL}}$ ——多路数据转换器控制信号，当它为低电平有效状态时，控制数据流向不进入系统总线。输入状态与 GA_8-8 相连。引腿号为 24。

D_7，D_6，D_5，D_4，D_3，D_2，D_1，D_0——系统数据总线上的数据信号。输出状态。引腿号为 25，26，27，28，29，30，31，32。

B_6，B_5，B_4，B_3，B_2，B_1，B_0——键盘输出的数据信号。输入状态。与键盘插座 KBSLOT-7，10，6、12，8，9，11 相连。引腿号分别为 33，34，35，36，37，38，39。

VDD——为 GA_2 器件的“漏”极，接 +5V 电源。引腿号为 40。

§ 2.3 动态随机存储器 RAM 行、列地址发生电路 GA_4

GA_4 门阵列芯片采用了半导体CMOS 工艺制作而成，主要的作用是产生动态随机存储器 RAM（4164）所需要的行、列地址。该器件要求单一+5V 电源。其引腿定义如图 2-3 所示。

A_3，A_2，A_1，A_0——从系统地址总线输入的地址信

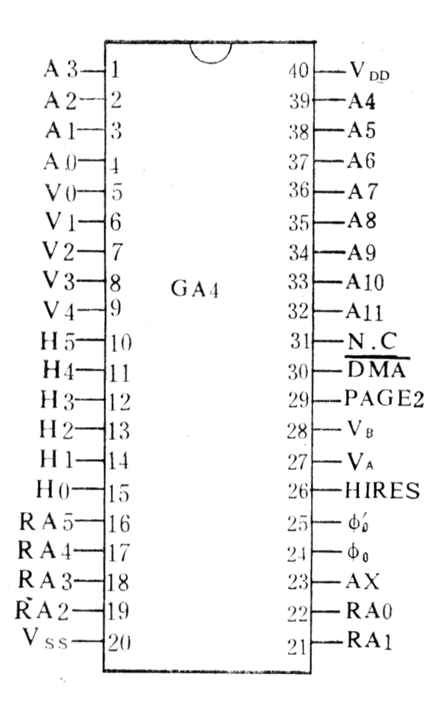


图 2-3 GA4 引腿定义图

号。输入状态。引腿号分别为 5， 6， 7， 8， 9。

H_5，H_4，H_3，H_2，H_1，H_0——视频水平计数信号。从 GA_1 送来。输入状态。引腿号分别为 10，11，12，13，14，15。

RA_5，RA_4，RA_3，RA_2，RA_1，RA_0——动态随机存储器 RAM（4164）的行、列地址信号。输出状态。送往 RAM4164 的 5， 6， 7， 12， 11， 10 引腿。它们的引腿号分别为 16， 17， 18， 19， 21， 22。

VSS——为 GA4 器件的“源”极接地，引腿号为20。

AX——总线地址与视频地址分选电路的控制信号。输入状态，与 IC 器件 Z42-14 引腿相接，引腿号为 23。

Φ0——1MHz 系统时钟。输入状态，与 IC 器件 Z35-7 相接。引腿号为 24。

Φ0′——送往 65SC02 CPU 所需时钟 1MHz，输出状态。与 65SC02 CPU 37 引腿相接。它的引腿号为 25。

HIRES——视频显示分辨率选择信号，当该信号为高电平时选择高分辨率方式，当为低电平时选择低分辨率方式。由 GA5-34 输出该信号。在 GA4 中它处于输入状态，引腿号为 26。

VA，VB——视频显示垂直计数信号。由 GA1 的 35 及 8 分别输出该信号。它在 GA4 中的引腿号为 24 及 25。

PAGE2——视频显示页面选择信号。由 GA5 的 32 引腿输出该信号，当为高电平时选择第二页面，当为低电平时选择第一页面。在 GA4 中为输入状态。引腿号为 29。

$\overline{\text{DMA}}$——直接存储请求信号。输入状态。低电平有效。腿引号为 30。

A11，A10，A9，A8，A7，A6，A5，A4——系统地址信息直接从系统地址总线输入。引腿号分别为 32，33，34，35，36，37，38，39。

VDD——为 GA4 器件的“漏”极，接+5V 电源。引腿号为 40。

GA4 器件的引腿 31 未用。

§ 2.4 视频显示方式控制电路 GA5

GA5 门阵列芯片采用了半导体CMOS 工艺制作而成，主要的作用是产生视频显示方式，如文本，图形，混合显示方式，高分辨率，低分辨率，第一页面，第二页面等显示方式的控制信号。该器件要求单一+5V 电源，包封在工业标准 40 条引腿双列直插式的管壳中。其引腿定义如图 2-4 所示。

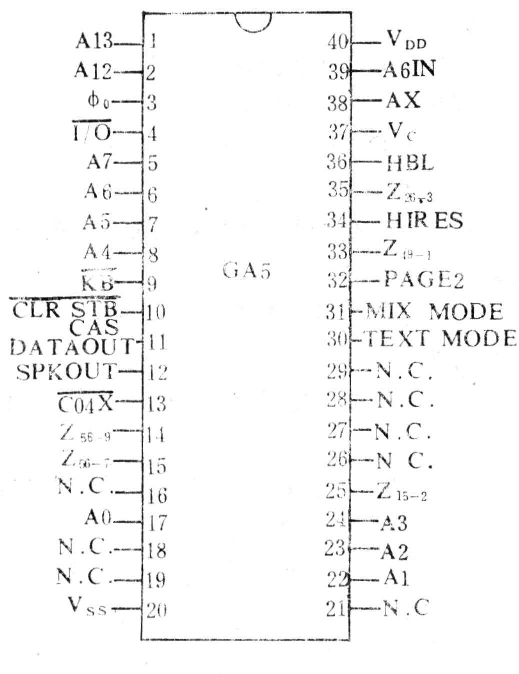


图 2-4 GA5 引腿定义图

A13，A12——从系统地址总线输入的地址信号。引腿号为 1 和 2。

Φ0——1MHz 系统时钟。输入状态。与 IC 器件Z35-7

相接。引腿号为3。

$\overline{C_0XX}$——I/O 译码信号，由 IC 器件 Z11-15 引腿送出。输入状态。引腿号为4。

A7，A6，A5，A4——从系统地址总线输入的地址信号。引腿号分别为5，6，7，8。

$\overline{KB}$——键盘数据输入允许信号。输出状态。低电平有效。引腿号为9。

$\overline{CLRSTB}$ 清键盘选通的控制信号。与GA2-21 相接。输出状态，低电平有效。引腿号为 10。

CAS DATA OUT——触发磁带盒转录输出的控制信号。输出状态。引腿号为 11。

SPKOUT——扬声器输出的控制信号。输出状态。引腿号为 12。

$\overline{C_{04}X}$——可扩充软开关控制信号。输出状态，低电平有效。引腿号为 13。

Z56-9——游戏接口控制信号。输出状态。低电平有效。引腿号为 15。

Z56-7——定时器 Z543a（558）的选通信号。输出状态，低电平有效。引腿号为 15。

A_0，A_1，A_2，A_3——从系统地址总线输入的地址信号。引腿号为 17，22，23，24。

VSS——GA5 器件的“源”极接地，引腿号为 20。

Z15-2　HIRES 信号的选通信号，由 GA1-10 送出。输入状态。低电平有效。引腿号为 25。

TEXT MODE——文本图形方式选通信号。输出状态，当输出为低电平时，选通图形方式，当输出为高电平时，

选通文本方式。引腿号为 30。

MIXMODE——字符与图形混合方式选通信号。输出状态。当输出为低电平时，选通非混合方式，当输出为高电平时，选通字符与图形混合显示方式。引腿号为 31。

PAGE2——视频显示页面选择信号。输出状态。当输出为低电平时，选择第一页面，当输出为高电平时，选择第二页面。引腿号为 32。

Z49-1——视频显示页面选择的控制信号。输出状态。直接与 IC 器件 Z48-9，Z49-1 及 R74-2 引腿相连。引腿号为 33。

HIRES——视频显示分辨率选择信号。输出状态。当输出为低电平时，选择低分辨率显示方式，当输出为高电平时，选择高分辨率显示方式。引腿号为 34。

Z26-3——组合逻辑控制信号。输出状态，引腿号为 35。在系统中未使用。

HBL — 视频行消隐控制信号。输入状态。为一个正脉冲。引腿号为 36。

VC——视频垂直计数信号。输入状态。引腿号为 37。

AX——总线地址与视频地址分选电路的控制信号。输入状态，与 IC 器件 Z42-14 引腿相接。引腿号为 38。

A6IN——产生 RA6 行、列地址的信号。输出状态。引腿号为 39。

VDD——为GA5 器件的“漏”极，接 +5V 电源。引腿号为 40。

该器件的 16，18，19，21，26，27，28，29 共八条引腿未使用。

§ 2.5 汉字、图形方式选择电路 GA7

GA7 门阵列芯片采用了半导体 CMOS 工艺制作而成，主要作用是产生汉字系统所需控制信号，存放监控程序，BASIC 程序的 ROM27256 的高位地址，某些选通信号及 RAM 行、列地址 RA6，RA7 的信号。该器件要求单一+5V 电源，包封在工业标准 40 条引腿双列直插式管壳中。其引腿定义如图 2-5 所示。

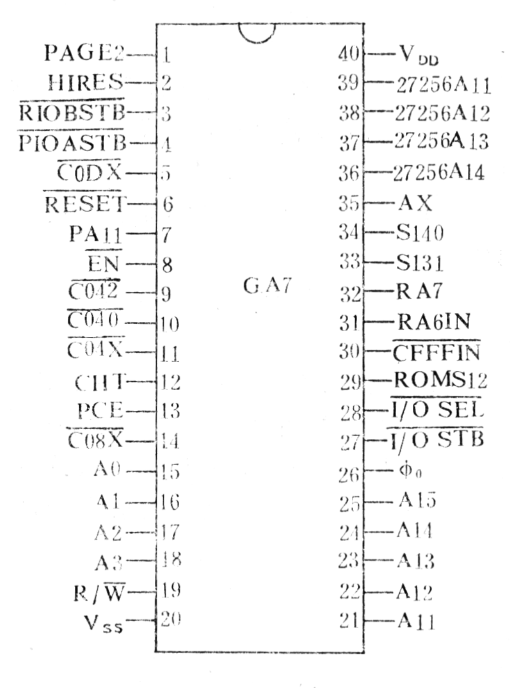


图 2-5 GA7 引腿定义图

PAGE2——视频显示页面选择信号。由 GA5-32 输出直接接到 GA7-1。输入状态。在 GA7 中该信号与 $C04F 组合产生第三、四高分辨率图形页。当$C04F 为高电平时，若 PAGE2 为低电平选择第三高分辨率图形页；若 PAGE2 为高电平，则选择第四高分辨率图形页。引腿号为 1 。

	$C04F/$C04E	PAGE2
第三图形页	1	0
第四图形页	1	0

HIRES——视频显示分辨率选择信号，从 GA5-34 输出。输入状态，低电平有效。引腿号为 2 。

$\overline{\text{PIOBSTB}}$——Z80 PIO B 口选通信号。输入状态。低电平有效，引腿号为 3 。

$\overline{\text{PIOASTB}}$——Z80 PIO A 口选通信号。输入状态。低电平有效，引腿号为 4 。

$\overline{\text{CODX}}$——译码信号。由 IC 器件 Z116-10 送出，输入状态。低电平有效。引腿号为 5 。

$\overline{\text{RESET}}$——系统复位信号。输入状态。引腿号为 6 。

PA11——为 ROM2764 高位地址产生的选通信号。输出状态。引腿号为 7 。

$\overline{\text{EN}}$——打印机引导程序选通信号。从 GA8-39 送出，输入状态。引腿号为 8 。

$\overline{\text{C042}}$——发声选通信号。输出状态。低电平有效。引腿号为 9 。

$\overline{\text{C040}}$——送往游戏机的实用选通信号。输出状态。低电平有效。引腿号为 10。

$\overline{C_{04}X}$——扩展软开关的触发信号。与 GA_5-13相接。输入状态。引腿号为 11。

CHT — 汉字单色状态选择信号。输出状态。当该信号为低电平时，选择汉字并打开彩色，当为高电平时，选择汉字并关掉彩色。引腿号为 12。

PCE——为 ROM 2764 提供选通控制信号。输出状态。引腿号为 13。

$\overline{C_{08}X}$——译码信号。输入状态，与 IC 器件 Z116-15 相接。引腿号为 14。

A_0，A_1，A_2，A_3，A_{11}，A_{12}，A_{13}，A_{14}，A_{15}——从系统地址总线输入的地址信息。引腿号为 15，16，17，18，21，22，23，24，25。

$R/\overline{W}$ 读或写信号。输入状态。与 GA_8-11 相接。引腿号为 19。

VSS—— GA_7 器件的“源”极接地，引腿号为 20。

Φ_0——1MHz 系统时钟。输入状态。与 IC 器件 Z35-7 相接。引腿号为 26。

$\overline{I/OSTB}$——公用扩展接口地址（C_{800} 为起始地址的 2K 空间）。输出状态。引腿号为 27。

$\overline{I/OSEL}$——扩展接口槽号口地址。七个可扩展使用槽号口地址（1号～7号），分别为 C_1XX～C_7XX。输出状态。引腿号为 28。

$\overline{ROMS_{12}}$——EPROM 27256（Z117）的选通信号。引腿号为 29。

$\overline{CFFFIN}$——扩展槽口选通控制信号。当选择某号扩展槽号时，发此信号封住其它扩展槽口，并将选中扩展槽号

扩展 2K I/O 地址投入使用。引腿号为 30。

RA6IN——产生动态随机存储器 RAM (4164) 行、列地址 RA6 的逻辑控制信号。输出状态，与 GA5-39 A6IN 组合产生 RA6 行、列地址信息。引腿号为 31。

RA7——动态随机存储器 RAM (4164) 行、列地址信号。输出状态。引腿号为 32。

S131——RAM (4164) 列选 $\overline{\text{CASOUT}}$ 信号的选通信号。输出状态，直接与 GA8-17 相接。引腿号为 33。

S140——EPROM (27256 Z117) 选通的控制信号，直接与 IC 器件 Z111-5 相接。引腿号为 34。

AX——总线地址与视频地址分选电路的控制信号。输入状态，与 IC 器件 Z42-14 引腿相接。引腿号为 35。

27256A14，27256A13，27256A12，27256A11——EPROM (Z117 27256) 的高位地址信号。输出状态。引腿号分别为 36，37，38，39。

VDD——GA7 器件的“漏”极，接 +5V 电源。引腿号为 40。

§ 2.6 电视（彩色 PAL 制式）接口电路 GA8

GA8 门阵列芯片采用了半导体 CMOS 工艺制作而成。主要作用是产生 PAL 制式电视机接口所需的各种控制信号，及其它选通、地址等信号。该器件要求单一 +5V 电源，包封在工业标准 40 条引腿双列直插式管壳中。其引腿定义如图 2-6 所示。

A0，A7——从系统地址总线输入的地址信息。引腿号分别为 1，19。

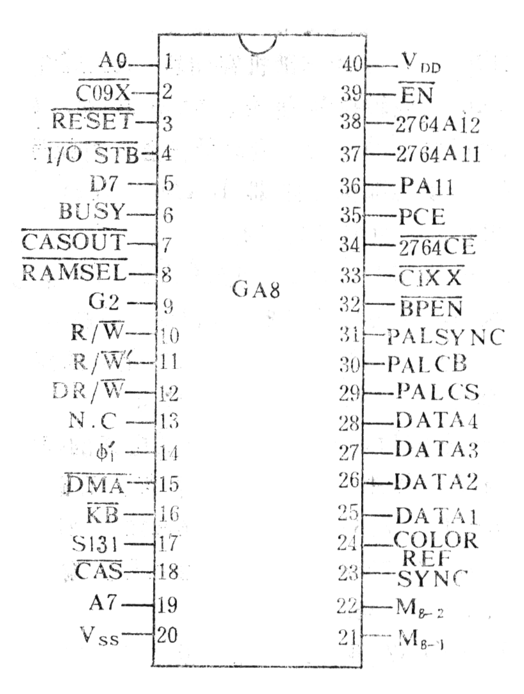


图 2-6 GA8 引腿定义图

$\overline{\text{C09X}}$ ——打印机启动控制信号。输入状态。与 IC 器件 Z116-4 相接。引腿号为 2 。

$\overline{\text{RESET}}$ ——系统复位信号。输入状态。低电平有效。引腿号为 3 。

$\overline{\text{I/O STB}}$——公用扩展接口地址($C800 为起始地址的 2K 空间)。输出状态。引腿号为 4 。

D7——从系统数据总线中来的数据信息，在此用于检测打印机“忙”信号状态。输出状态。信号引腿为 5 。

$\overline{\text{CASOUT}}$——动态随机存储器 RAM（4164）列地址选通控制信号。输出状态。低电平有效。引腿号为 7 。

BUSY——打印机“忙”状态信号。输入状态，直接由打印机插座 PRINTER SLOT-10 引腿送来。引腿号为 6。

$\overline{\text{RAMSEL}}$——多路数据转换器控制信号，当它为低电平有效状态时，控制数据流向不进入系统数据总线。输出状态。引腿号为 8 。

G2——控制地址总线驱动器 Z22,Z21（74LS244）选通信号。输出状态。引腿号为 9 。

R/$\overline{\text{W}}$——65SC02 CPU 发出的读写信号。输入状态。引腿号为 10。

R/$\overline{\text{W}}$′——为系统提供的读写信号。输出状态。引腿号为 11。

DR/W——为动态随机存储器 RAM（4164）提供的读写信号。输出状态。引腿号为 12。（第 13 引腿未用）

Φ1——由系统发出的 1MHz 时钟信号。输入状态。引腿号为 14。

$\overline{\text{DMA}}$——直接存储器存取请求信号。低电平有效。输入状态。引腿号为 15。

$\overline{\text{KB}}$——键盘数据输入允许信号。输入状态。低电平有效。引腿号为 16。

S131——RAM（4164）列选 $\overline{\text{CASOUT}}$ 信号的选通信号。引腿号为 17。

$\overline{\text{CAS}}$——列选通信号。低电平有效。输入状态。引腿号为 18。

A7——从系统地址总线输入的地址信号。引腿号为 19。

VSS——GA_8 器件的"源"极接地，引腿号为 20。

SYNC′——同步信号。由 GA_{1-19} 发出，经信号处理由 D_{101-2} 送往 GA_{8-23}。输入状态。引腿号为 23。

COLOR REF——3.5MHz 彩色基振频率。输入状态。由时序电路 Z_{35-3} 发出。引腿号为 24。

$DATA_1$，$DATA_2$，$DATA_3$，$DATA_4$——视频数据信号。由 IC 器件 Z_{114} 发出。输入状态。信号引腿为 25，26，27，28。

PALCS——PAL 制彩色副载频色度信号。直接与 $74LS_{27}$ 三输入或非门的一个输入端相接。输出状态。引腿号为 29。

PALCB——PAL 制彩色同步信号，输出状态，直接与 IC 器件 $74LS_{27}$ 三输入端正或非的一个输入端相接。引腿号为 30。

PALSYNC——PAL 制彩色同步信号。输出状态。直接与 R_{207} 相接，引腿号为 31。

$\overline{BPEN}$——汉字或磁盘操作时发出的选通 EPROM 2764 的请求信号。输入状态，直接与 IC 器件 Z_{58-8} 引腿相接。引腿号为 32。

$\overline{C_1XX}$——打印机操作时发出的选通 PEROM 2764 的请求信号，低电平有效。输入状态。引腿号为 33。

$\overline{2764CE}$——EPROM 2764 选通信号。输出状态。低电平有效。引腿号为 34。

PCE——2764CE 信号的控制信号。输入状态，直接与 GA_{7-13} 相接。引腿号为 35。

PA_{11}——$2764A_{11}$ EPROM2764 地址信号产生的控

制信号。输入状态。直接与 GA7-7 相接。引腿号为 36。

2764A11——EPROM 2764 的高位地址 A11。输出状态。引腿号为 37。

2764A12——EPROM 2764 的高位地址 A12。输出状态。引腿号为 38。

$\overline{EN}$ ——打印机操作发出的允许信号。输出状态。低电平有效。引腿号为 39。

VDD——为 GA8 器件的“漏”极，接+5V 电源。引腿号为 40。

第三章　系统时钟信号发生器和水平时序

计算机系统是按照时钟分时顺序控制的，因此时序是计算机系统正常运行的基本依据。它直接影响着计算机的运行和处理速度。

§3.1　时钟信号

XMF-1 微型计算机的时序电路如图 3-1 所示。它是由四 D 触发器 Z35（74LS175）、移位寄存器 Z42（74LS195）和双 4 线选 1 的数据选择器 Z33(74LS153)组成。产生系统所要求的时钟信号是：

主振频率　14.3 MHz。

7MHz　7.2 MHz。

$\overline{\text{7MHz}}$　7.2 MHz。

$\overline{\text{RAS}}$　RAM 地址行选通信号 2MHz。

AX　地址多路转换信号 2MHz。

$\overline{\text{CAS}}$　RAM 地址列选通信号 2MHz。

Q3　供外围接口用的通用时钟信号 2MHz。

Φ0　为系统提供的 1MHz 振荡频率的时钟信号。

Φ1　为Φ0 的反相信号 1MHz。

$\overline{\text{LDPS}}$　视频地址发生器触发时钟及文本方式送数据

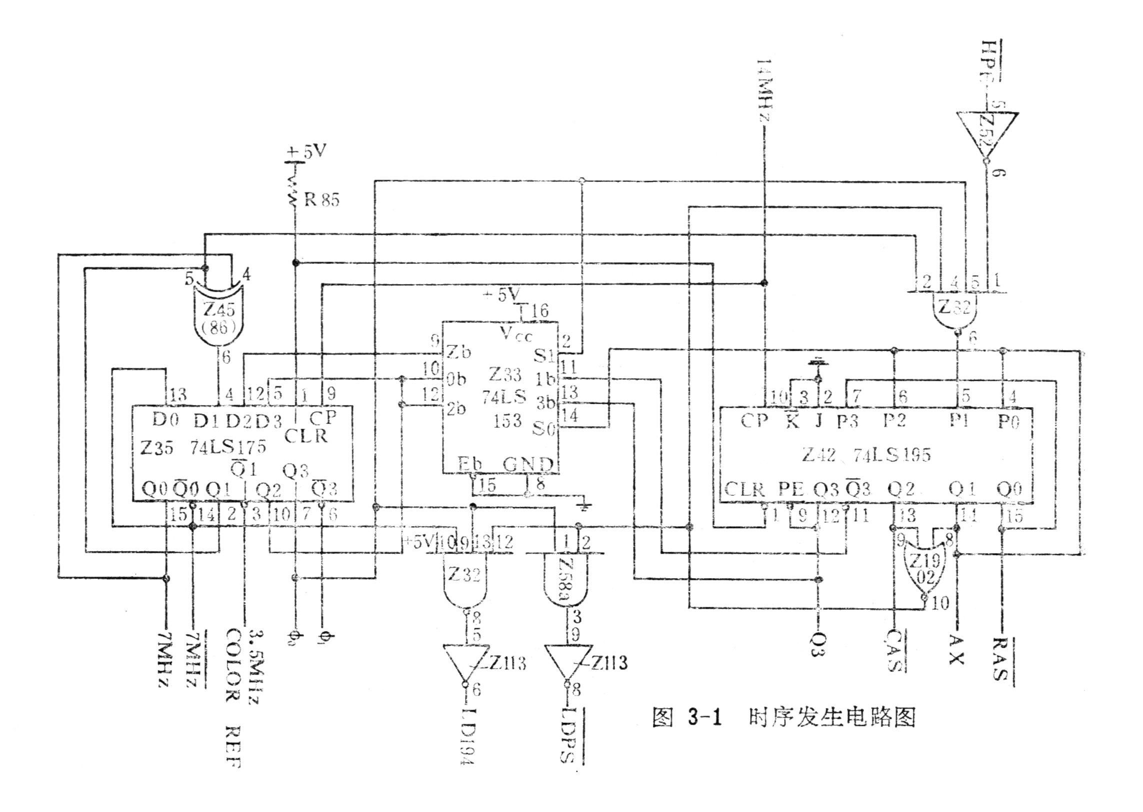

+5V
R 85
Z45
(86)
Z35 74LS175
D0 D1 D2 D3 CP CLR
Q0 $\overline{Q0}$ Q1 $\overline{Q1}$ Q2 Q3 $\overline{Q3}$
7MHz
$\overline{7MHz}$
3.5MHz
COLOR REF
φ0
φ1
Z33 74LS 153
Vcc S1 1b 3b S0 GND Eb Zb 0b 2b
+5V
Z32
Z58a
Z113
LD194
LDPS
14MHz
HPE
Z52
Z32
Z42 74LS195
CP K J P3 P2 P1 P0
CLR PE Q3 $\overline{Q3}$ Q2 Q1 Q0
Z19 02
Q3
$\overline{CAS}$
AX
$\overline{RAS}$

图 3-1 时序发生电路图

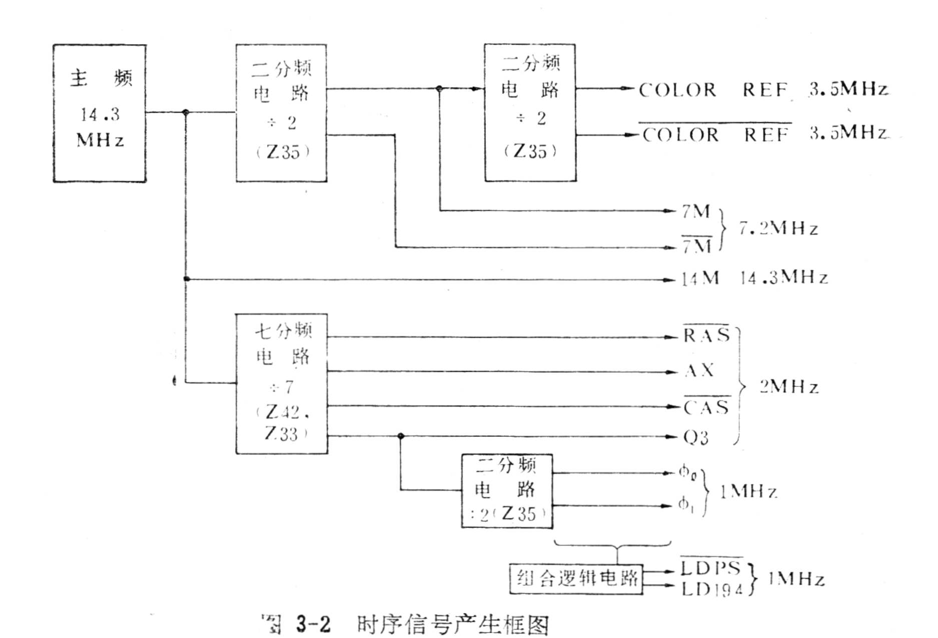


图 3-2 时序信号产生框图

的控制信号 1MHz。

LD194 由 AX, CAS, 7MHz 及 Φ0 组合产生的信号，用于输送图形数据。

经主振时钟（14.3MHz）分频后产生的这些时钟信号框图，如图 3-2 所示。

§ 3.2 主振时钟（14.3 MHz）发生电路

“小蜜蜂-I”微型计算机的主振时钟电路如图 3-3 所示。根据 NTSC 制式彩色电视机的彩色信号负载频 3.579545MHz 的精确要求，考虑到系统分频电路设计与实现的可能性，系统主振频率选取彩色信号负载频的 4 倍，即：

$$4 \times 3.599545\text{MHz} = 14.31818\text{MHz}。$$

主振时钟电路中的主振信号是由一个晶体振荡器产生的，其谐振频率为 14.31818MHz，经整形驱动后形成方波输出。如图 3-4 中 14MHz 标志行所示方波波形。

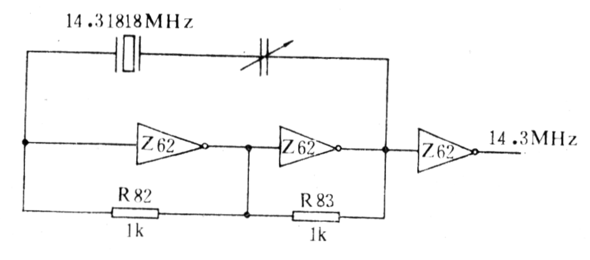


图 3-3 主振时钟振荡电路

由时序发生器电路产生的系统时序图如图 3-4 所示。分析时序发生电路（图 3-1）可知，用四 D 触发器 Z35 的一组 D 触发器作为二分频，产生 7MHz（Z35-15 引腿输出）和 $\overline{7\text{MHz}}$（Z35-14 引腿输出）两个时钟信号。然后利

用 Z35 的另一组 D 触发器，对 7MHz 再进行二分频，产生 3.579545MHz（Z35-3 引腿输出）彩色参考时钟信号（COLOR REF）。

时钟Q_3，RAS，AX，CAS 信号是利用移位寄存器 Z42（74LS195）的移位与并行接收的特性产生的。IC74LS195 器件的第 9 引腿（Z42-9）是工作方式选择端，当它为低电平时，工作在并行接收操作方式；为高电平时，工作在数据移位操作方式，此时输出端 Q_0、Q_1、Q_2、Q_3 在 14.3 MHz 主振脉冲上升沿的作用下，由串行输入端 J、K→Q_0，由 Q_0→Q_1，Q_1→Q_2，Q_2→Q_3 依次将数据移位传送。

若最初状态 Q_3 在主振前沿作用下为低电平（如图 3-4 所示），此时由于 Q_3（Z42-12）与PE（Z42-9）相接，所以使 Z42 器件处于并行接收工作方式，因此下一个脉冲时钟到来时，输出端 Q_0、Q_1、Q_2、Q_3 的数据将呈现出对应输出端 P_0、P_1、P_2、P_3 的数据，假设 P_1 为高电平，输出端 Q_0、Q_1、Q_2、Q_3 对应值为 0100，由于 Q_0 与 P_3，Q_1 与 P_0 和 P_2 相接，所以此时 P_0、P_1、P_2、P_3 值如为 1110，在下一个主振脉冲到来时，利用其上升沿使 Q_0、Q_1、Q_2、Q_3 为 1110。根据同样道理分析，在下一个脉冲上升沿作用下，使其值为 1111。由于 Q_3 为高电平，迫使 PE 为高电平，此时 Z42 改变工作方式，由并行接收转为串行移位。在下一个脉冲上升沿作用下，Z42-2、3 串行数据输入端 J、K 的低电平移入 Q_0，Q_1、Q_2、Q_3 是相应 Q_0、Q_1、Q_2 的值，所以 Q_0、Q_1、Q_2、Q_3 为 0111，这样一直要维持连续 4 个脉冲作用，才能使 0 移入 Q_3，再转换到并行接收工作方式。其操作真值表如表 3-1。

Φ0 和 Φ1 系统时钟是用 Q3 二分频产生 1MHz 振荡频率。它是利用双 4 选 1 数据选择器（74LS153）Z33 与 D 触发器（74LS175）Z35 组成逻辑电路，由 Z33-9 引腿与 Z35 第三组触发器产生 1MHz 信号。由 Z35-7 引腿输出。

表 3-1

	P0	P1	P2	P3	Q0	Q1	Q2	Q3	
初始状态					0	0	0	0	并行接收方式
第一个脉冲	0	1	0	0	0	0	0	0	
第二个脉冲	1	1	1	0	0	1	0	0	
第三个脉冲	1	1	1	1	1	1	1	0	
第四个脉冲	1	1	1	1	1	1	1	1	
第五个脉冲	1	1	1	0	0	1	1	1	串行移位方式
第六个脉冲	0	1	0	0	0	0	1	1	
第七个脉冲	0	1	0	0	0	0	0	1	
第八个脉冲	0	1	0	0	0	0	0	0	
第九个脉冲	1	1	1	0	0	1	0	0	并行接收方式

Z33 器件是由 S1、S0 信号控制进行多路传输转换的，即：00 选择0b 通道，01 选择 1b 通道，10 选择2b 通道，11 选择3b 通道。当 Z35-7 和 Z35-10 引腿为低电平时，AX、Φ0 都为低电平时（请参阅图 3-3），S0 和 S1 为低电平，Z33-9 输出端选通的是Z33-10 引腿的低电平，在下一个脉冲触发下，使 Z35-10 为低电平。当 AX 变为高电平，Φ0 仍为低电平时，选中 Z33-11 引腿上的高电平送入 Z35-12，在下一个脉冲触发下使 Z33-10 为高电平，同时该状态又反馈到 Z35-5 第四组 D 触发器的输入端及Z33-10 和 Z33-12 引腿，在下一个脉冲触发下，不但 Z35-10 为高

电平，同时 Z35-5 的高电平也打入 Z35-7，使 Φ0 变为高电平。以后 AX、Φ0 都为高电平，Z33-9 选通的为 3b 通道，Z33-13 引腿的高电平送入 Z35-12，保持 Φ0 为高电平状态。当进行到 Q3 变为低电平，AX、Φ0 仍为高电平时，将选中 3b 通道，将低电平送入Z35-12，在下一个脉冲触发下，使 Z35-10 引腿变为低电平，并将该状态反馈给 Z35-5 和 Z33-10 以及 Z33-12 引腿，在下一个脉冲触发下，使 Φ0 变为低电平。在以后的 7 个 14MHz 脉冲触发作用下，都使 Φ0 为低电平状态，从而完成 Φ0 的一个周期。

$\overline{\text{LDPS}}$ 和 LD194 时钟信号是由 Z58a、Z32、Z113形成组合逻辑电路产生的。只有当 AX 和 $\overline{\text{CAS}}$ 为低电平，Φ0 为高电平时，$\overline{\text{LDPS}}$ 才为低电平；只有当 $\overline{\text{7MHz}}$、Φ0 为高电平，AX，$\overline{\text{CAS}}$ 为低电平时，LD194 才为高电平。

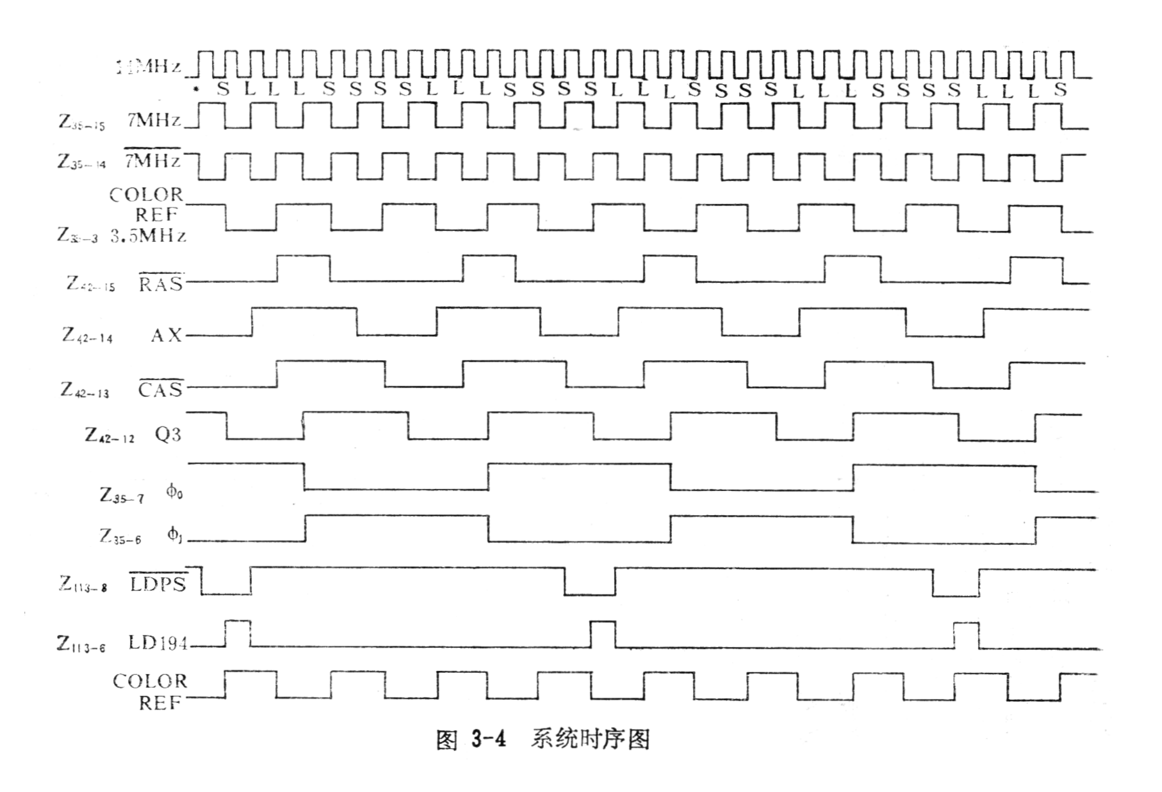
由时序发生器电路产生的系统时序图如图 3-4 所示。

这个时序图与实际时序图尚有一些不同，主要是因为图形显示时要求有一段时间的延迟。我们将在介绍完水平时序后就会弄清这部分的要求。

§ 3.3 水平时序

视频地址是 CRT 访问存储器 RAM 时需要的地址，其地址计数信号是 H0，H1，H2，H3，H4，H5，$\overline{\text{HPE}}$，VA，VB，VC，V0，V1，V2，V3，V4，V5 共 16 条视频地址线。

由于一帧屏幕水平行要求显示 40 个字符，行回扫时间要求有 25 个字符显示的时间长度。所以一行共要求有 65 个



MHz
S L L L S S S S L L L S S S S L L L S S S S L L L S S S S L L L S
Z35-15 7MHz
Z35-14 7MHz
COLOR REF
Z35-3 3.5MHz
Z42-15 RAS
Z42-14 AX
Z42-13 CAS
Z42-12 Q3
Z35-7 φ0
Z35-6 φ1
Z113-8 LDPS
Z113-6 LD194
COLOR REF

图 3-4 系统时序图

时间计数，才能重新开始下一行字符显示。

一帧屏幕要求垂直行有 192 个显示行，帧消隐回扫时间要求有相当于 70 个字符显示时间的计数，才能从一帧的底部最后一个字符位置回到一帧最初的一个字符位置，垂直计数总共需 262 个计数。

用 H_0，H_1，H_2，H_3，H_4，H_5，$\overline{HPE}$ 组成水平计数信息，最多可计数到 2 的 7 次方 128 个计数，但实际只需要 65 个。

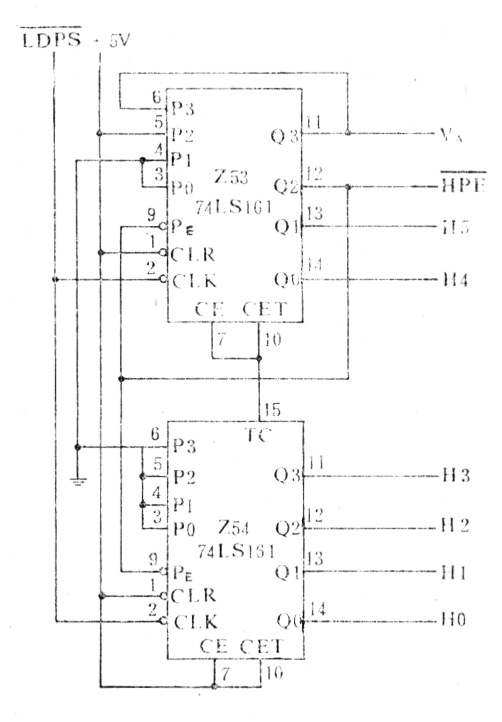


图 3-5 水平时序发生电路

用 VA, VB, VC, V_0, V_1, V_2, V_3, V_4, V_5 九个信号产生垂直计数，总共可形成 2 的 9 次方共 512 个计数，但实际只需要 262 个计数，即可满足垂直计数的要求。

现在我们仅就水平视频地址的形成及如何产生时序延迟问题进行讨论。

水平视频计数(即水平视频地址)电路是由 2 个74LS161 计数器完成的。其电路图如图 3-5 所示。实际电路已将此电路做入门阵列 GA_1 中。

利用 74LS161 计数器的并行数据接受和计数工作方式，使它的控制端 PE (Z_{54}-9) 受到 $\overline{HPE}$ 的控制，当进入初始状态后，在 $\overline{LDPS}$ 第一个脉冲到来时，使 Z_{54}-9 处于计数状态，当计满 16 个数时，向 Z_{53} 进位，使 Z_{53} 计一次数，然后 Z_{54} 又回到 0 开始计数。计满 16 个数再进位使 Z_{53} 计数，如此在计完 64 次后，$\overline{HPE}$ 再次降为 0，使 Z_{53}、Z_{54} 又处于计数状态，此时为第 65 个计数，输出端为 0，$\overline{HPE}$ 降为 0 是产生时序延长的主要因素。其时序图，如图 3-6 所示。

在水平第 65 个计数时，图 3-1 中的 Z_{32}-6 降为低电平，其逻辑表达式为：

$$Z_{32}\text{-}6=\overline{\overline{HPE}\cdot AX+\overline{CAS}\cdot\Phi_0\cdot\overline{COLORREF}}$$

进一步分析会得到

$$Z_{32}\text{-}6=\overline{\overline{HPE}\cdot LDPS\cdot COLORREF}$$

从时序图 3-7 中第 65 个计数可以看到，由 Z_{32}-6 连续向 AX 送出两个 14MHz 时钟周期，因而使 $\overline{RAS}$、$\overline{CAS}$，Q3，Φ0，Φ1 和 $\overline{LDPS}$ 也都延长两个 14MHz 时钟周期。直到 $\overline{COLORREF}$ 升为高电平，才可能在下一个时钟到来

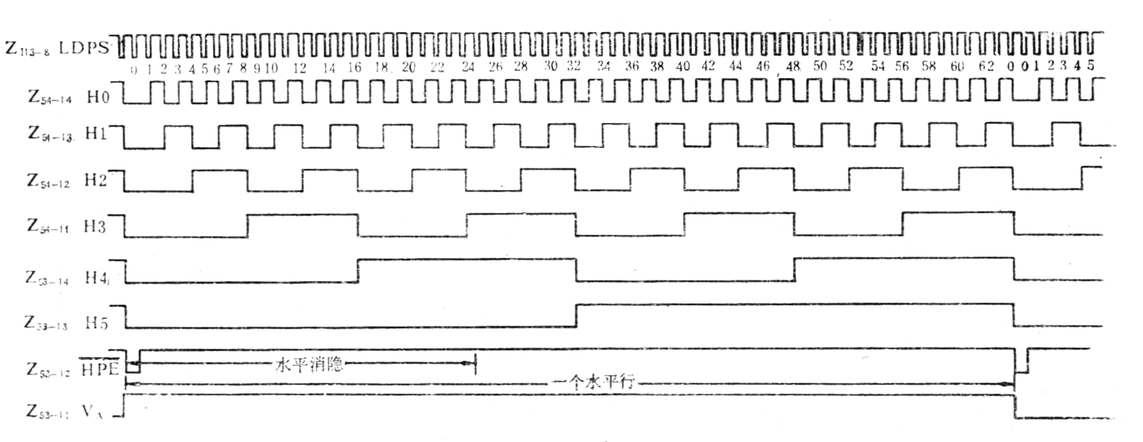

Z113-8 LDPS
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 0 0 1 2 3 4 5
Z54-14 H0
Z54-13 H1
Z54-12 H2
Z54-11 H3
Z53-14 H4
Z53-13 H5
Z52-12 HPE
水平消隐
一个水平行
Z53-11 VA

图 3-6 视频水平时序

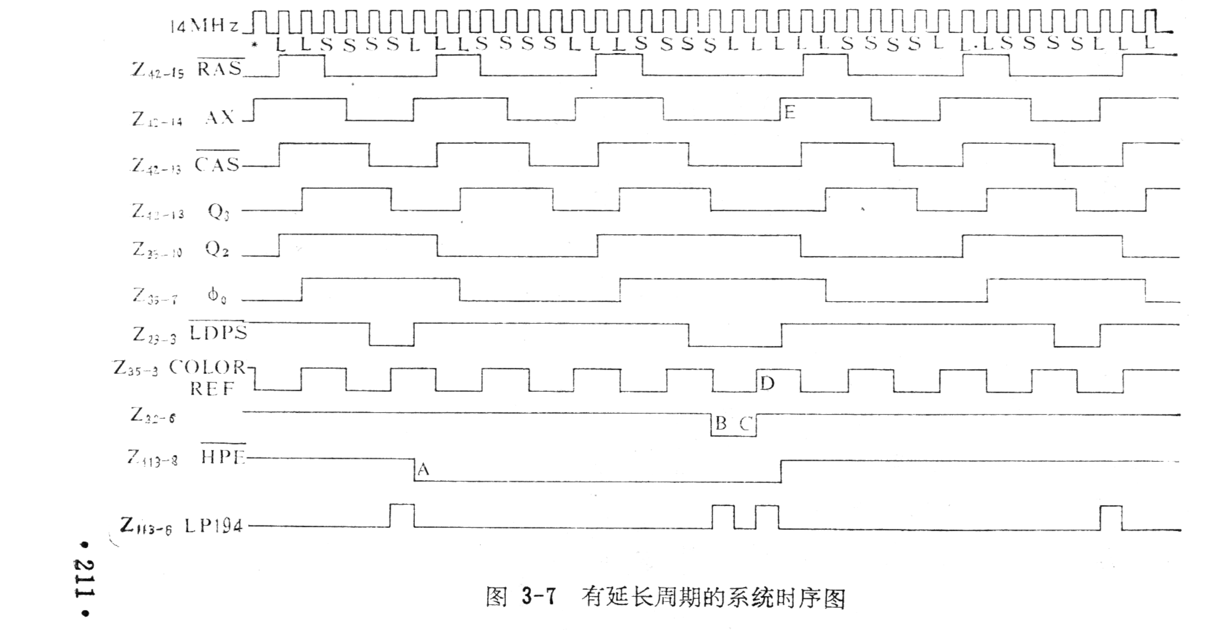


图 3-7 有延长周期的系统时序图

时使 AX 变为高电平，时序延迟停止。具有延迟周期的时序如图 3-7 所示。

一般来说正常的 64 个计数中每个时钟周期的时间应为：

$$14 \div 14.31818\text{MHz} = 976\text{ns}$$

在第 65 个延长周期应为：

$$16 \div 14.31818\text{MHz} = 1117\text{ns}$$

其平均速度应为：

$$14.31818\text{MHz} \times 65/((64 \times 14) + 16) = 1.020484\text{MHz}$$

其视频水平扫描率应是：

$$1.020484\text{MHz} \div 65 = 15.700\text{KHz}$$

这和彩色电视所要求的水平扫描频率 15.734KHz 非常接近，已能满足其要求。若采用 15.734KHz 作为视频水平扫描频率，可以看到对要求精度很高的色同步频率3.579545 MHz 的分频系数应为：

$$3.579545\text{MHz} \div 15.734\text{KHz} = 227.5$$

即每一行相位与前一行相差 180 度。为此取整数分频系数为 228，才产生了前述的时序延迟的情况。

第四章　视频显示系统

§4.1　概述

视频显示是微型计算机的主要输出方式之一。中华学习机“小蜜蜂-I”（XMF-I）充分考虑了我国的实际情况，设有两种输出信号：

1. NTSC制式的视频全电视信号，由标有MONITOR的插口输出，输出电压幅度不小于1Vp-p。

2. PAL制式的高频调幅全电视信号，由标有TV的插口输出，高频载波的中心频率约为203MHz（10频道）。

这两种输出信号中，前一种主要是为视频监视器所使用，后一种则为家用电视机所使用，PAL制的视频监视器亦可使用该信号。

在显示模式方面，XMF-I微型计算机存在五种显示模式，即：

1. 黑白文本模式（TEXT）。
2. 低分辨率图形模式（LORES）。
3. 高分辨率图形模式（HIRES）。
4. 文本和低分辨率图象混合显示模式。
5. 文本和高分辨率图象混合显示模式。

TEXT模式和LORES模式都有2页显示区，HIRES有4页显示区。这些显示模式和页面的转换均由屏幕软开关来完成。

图 4-1 给出整个视频显示系统的结构框图。

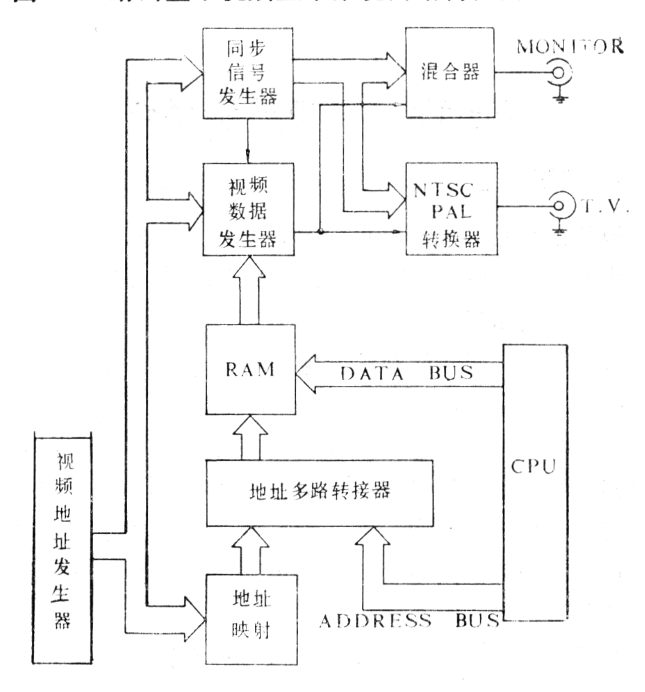


图 4-1 视频显示系统框图

§ 4.2 显示模式

1. 文本模式（TEXT）

文本模式允许屏幕上显示 24 行字符，每行 40 个字符宽。每个显示位置占有显示缓冲区内一个字节。显示缓冲区的地址范围如下：

第 1 页：$400～$7FF

第 2 页：$800～$BFF

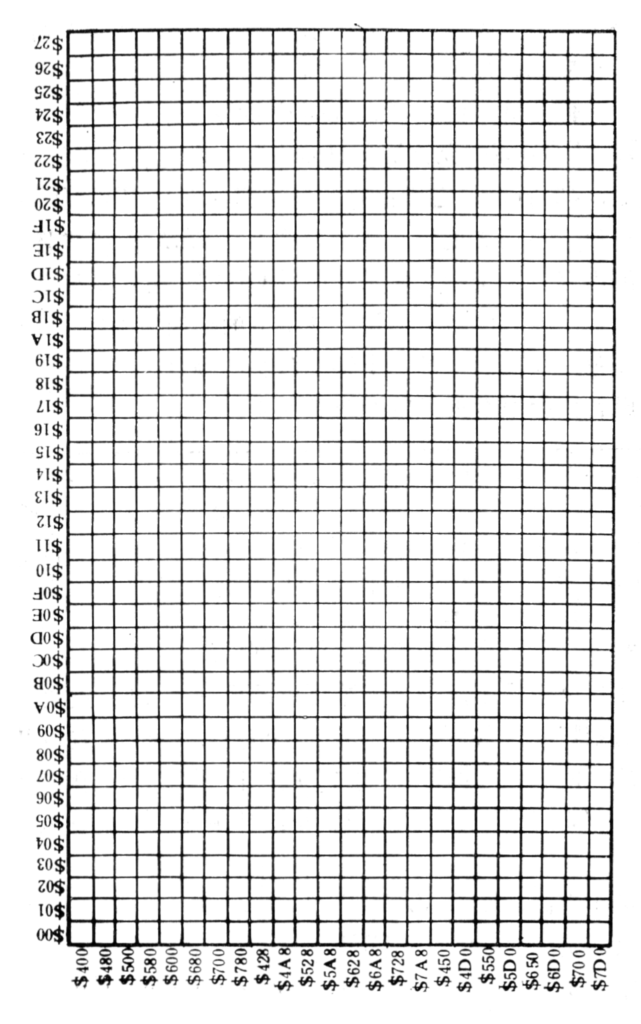


图 4-2 文本模式和低分辨率图形模式的显示缓冲区分配图

图 4-2 给出 TEXT 模式中显示缓冲区的地址分配图。图中每个方格代表在屏幕上显示的字符位置，纵横坐标之和表示了相应的缓冲区地址。

2. 低分辨率图形模式（LORES）

LORES 模式允许屏幕显示 1920 个色块，每行 40 个色块宽，48 行高。它所占有的显示缓冲区与 TEXT 模式完全一样，不同之处仅在显示方式上。TEXT 模式将每个字节的内容以字符形式显示，而 LORES 模式将每个字节显示成上下相邻的两个色块。该字节的低四位确定上半个色块的颜色，而高四位确定下半个色块的颜色。

颜色同半字节的值的对应关系是：

0——黑	4——深绿	8——棕	12——绿
1——品红	5——灰	9——橙	13——黄绿
2——深蓝	6——蓝	10——灰	14——浅绿
3——紫	7——浅蓝	11——粉红	15——白

3. 高分辨率图形模式（HIRES）

在这个模式中，整个屏幕可以看成一个 280 点宽、192 点高的矩阵，一共可显示 53760 个色点。每个色点占用显示缓冲区的一位（bit），每 7 位占用一个字节。因此，每个 HIRES 图形页占有 7680 个缓冲单元，其地址范围如下：

第 1 页：$2000～$3FFF

第 2 页：$4000～$5FFF

第 3 页：$6000～$7FFF

第 4 页：$8000～$9FFF

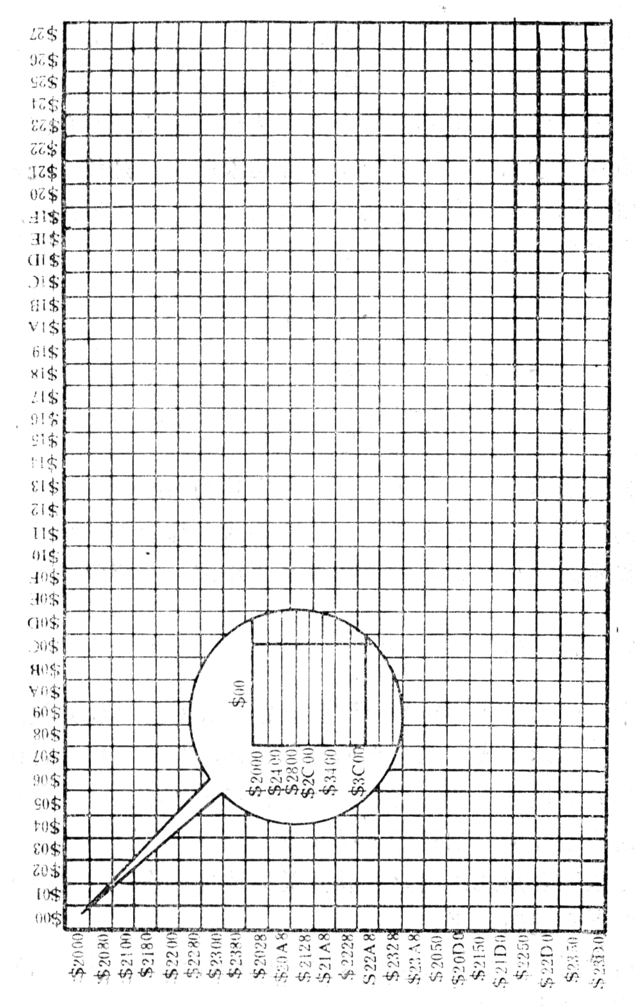


图 4-3 高分辨率图形模式的缓冲区分配图

图 4-3 给出了在 HIRES 第一页时，屏幕显示位置与缓冲区地址的对应关系。在每个缓冲区单元中，bit 7 不被显示，bit 0～bit 6 以从左到右的顺序显示在屏幕上。

在 HIRES 模式中，屏幕上能显示黑、白、紫、绿、橙、蓝六种颜色，但不是每个点都能显示这六种颜色。具体每个点显示什么色彩，受两个条件的限制：一是该点处在屏幕的哪一列上，二是该点所在字节的 bit 7 是 1 还是 0，这两个限制条件对色彩的影响见下表：

	bit 7=0	bit 7=1
奇数列	绿色	橙色
偶数列	紫色	蓝色

定义屏幕最左边为第 0 列，最右边为第 279 列。若显示点的值为 0，则显示黑色。若相邻两点都为 1，则这两点都将显示成白色，而不论这两点是否在同一字节中。

§ 4.3 屏幕软开关

中华学习机“小蜜蜂-I”共设置了五对屏幕软开关，从而对五种显示模式和八个页面进行转换。程序可以通过访问这些软开关地址，使得它们置于某种特定的状态。

这些软开关分别属于 GA5 和 GA7 两片门阵列电路。表 4-1 以列表形式给出这些软开关的地址、作用及其对应的管脚定义和逻辑电平。

通过对这五对软开关的组合，可以得到 14 种不同的显示方式，如表 4-2 所示。

表 4-1 屏幕软开关

地 址	作 用	对应管腿定义	逻辑电平
$C04E	显示第一页与第二页	在 GA7 内部	
$C04F	显示第三页与第四页*		
$C050	显示图形方式	TEXT MODE(GA5-30)	0
$C051	显示文本方式		1
$C052	显示全文本或全图象式方	MIX MODE (GA5-31)	0
$C053	显示混合方式		1
$C054	显示第一页或第三页	PAGE2 (GA5-32)	0
$C055	显示第二页或第四页		1
$C056	显示低分辨模式	在 GA5 内部	
$C057	显示高分辨模式		

* 仅对 HIRES 有效。

§ 4.4 视频地址及同步信号发生器

门阵列器件 GA1 为视频地址及同步信号发生器。H_0～H_5 各信号为水平地址信号，VA～VC 及 V_0～V_5 为垂直地址信号。另外还有几个同步信号。

1. 水平时序

水平时序已在系统时序一章内分析过，这里作一简单的

表 4-2 屏幕软开关的组合方式

显示模式	页 面	屏 幕 软 开 关
全文本	第 1 页	$C054, $C051
	第 2 页	$C055, $C051
全低分辨图形	第 1 页	$C054, $C056, $C052, $C050
	第 2 页	$C055, $C056, $C052, $C050
全高分辨图形	第 1 页	$C054, $C04E, $C057, $C052,$C050
	第 2 页	$C055, $C04E, $C057, $C052,$C050
	第 3 页	$C054, $C04F, $C057, $C052,$C050
	第 4 页	$C055, $C04F, $C057, $C052,$C050
文本与低分辨率图形混合	第 1 页	$C054, $C056, $C053, $C050
	第 2 页	$C055, $C056, $C053, $C050
文本与高分辨率图形混合	第 1 页	$C054, $C04E, $C057, $C053,$C050
	第 2 页	$C055, $C04E, $C057. $C053,$C050
	第 3 页	$C054, $C04F, $C057, $C053,$C050
	第 4 页	$C055, $C04F, $C057, $C053,$C050

回顾。

$\overline{LDPS}$ 信号送入 GA1 后，逐级分频产生 $H_0 \sim H_5$ 各水平地址信号。注意每隔 64 个$\overline{LDPS}$ 脉冲后，来一个 $\overline{HPE}$ 信号，同时产生一个扩展了的 $\overline{LDPS}$ 信号，同样，$H_0 \sim H_5$

各信号也在该点被扩展。不扩展的 $\overline{LDPS}$ 周期大约为978ns，扩展后的 $\overline{LDPS}$ 周期大约为 1117ns。因此，一个行周期大约为 63.7μs。

具体水平时序波形可参阅系统时序一章中的水平时序图。

2. 垂直时序

在水平时序中，每隔大约 63.7μs 产生一个 $\overline{HPE}$ 脉冲，即 $\overline{HPE}$ 信号的周期就是行周期。VA 信号是对 $\overline{HPE}$ 信号的二分频，即 VA 的周期为两个行周期。

VB～V5 是一个同步 8 位计数器，对 VA 进行同步计数。当计数到 11111111 时，该计数器进入预置状态，具体在下一个 VA 到来时被置成什么数，受 GA1 的第 38 号管脚（ACY1）控制。当 ACY1 为低电平时，将置成 01100100；当 ACY1 为高电平时，将置成01111101。若置成了01100100，则它计数计到 11111111，共有 156 个 VA周期，312 个行周期，因此它的场周期为 $312 \times 63.7\mu s$，约为 20ms，即 50Hz。

若置成 01111101，则为 60Hz 的场频。

50Hz 场频的垂直时序关系见图 4-4。由于受排版的限制，将此图截为 4 段，请读者注意。

3. 同步信号

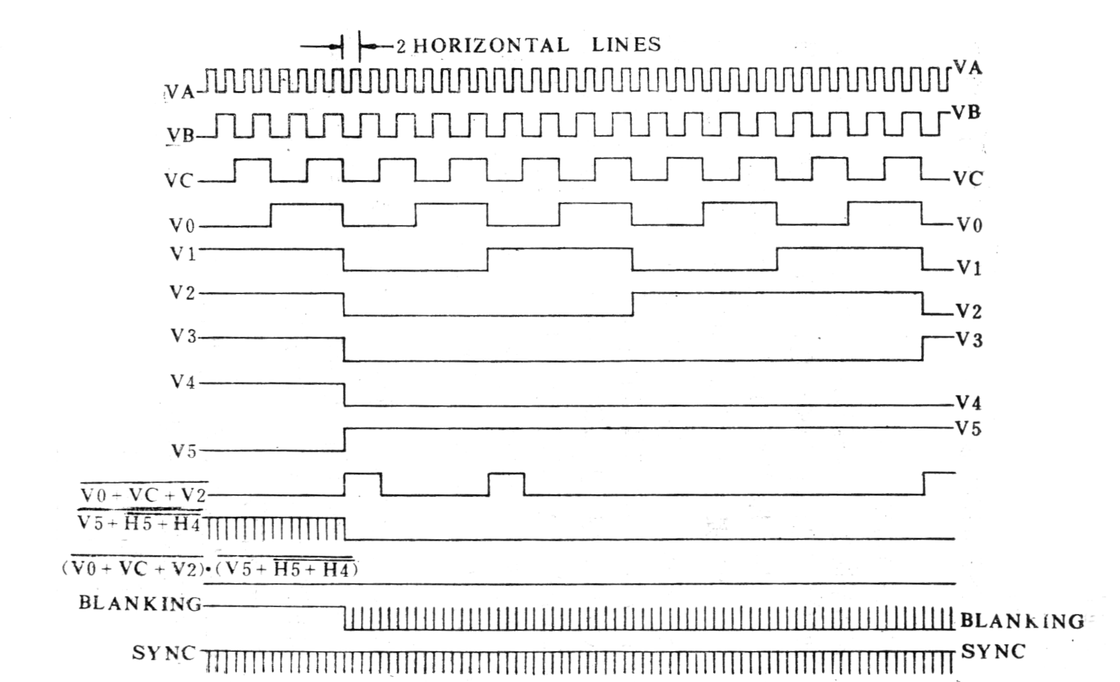
GA1 产生的同步信号有如下几个：

SYNC——同步信号；

HBL ——行消隐信号；

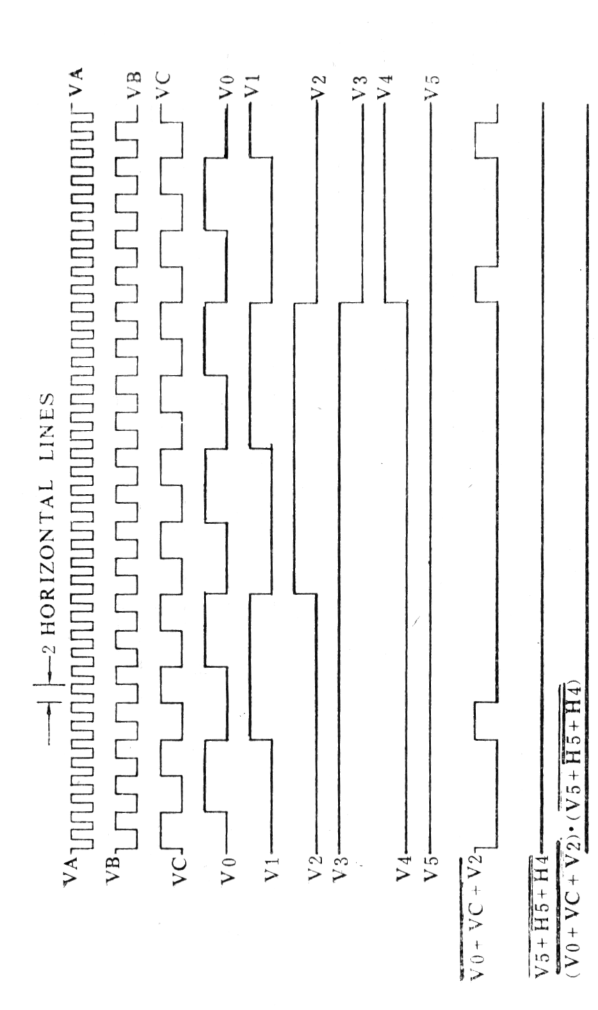
Z38-5 ——混合消隐信号（BLANKING）；

COLOR BURST——色同步信号。

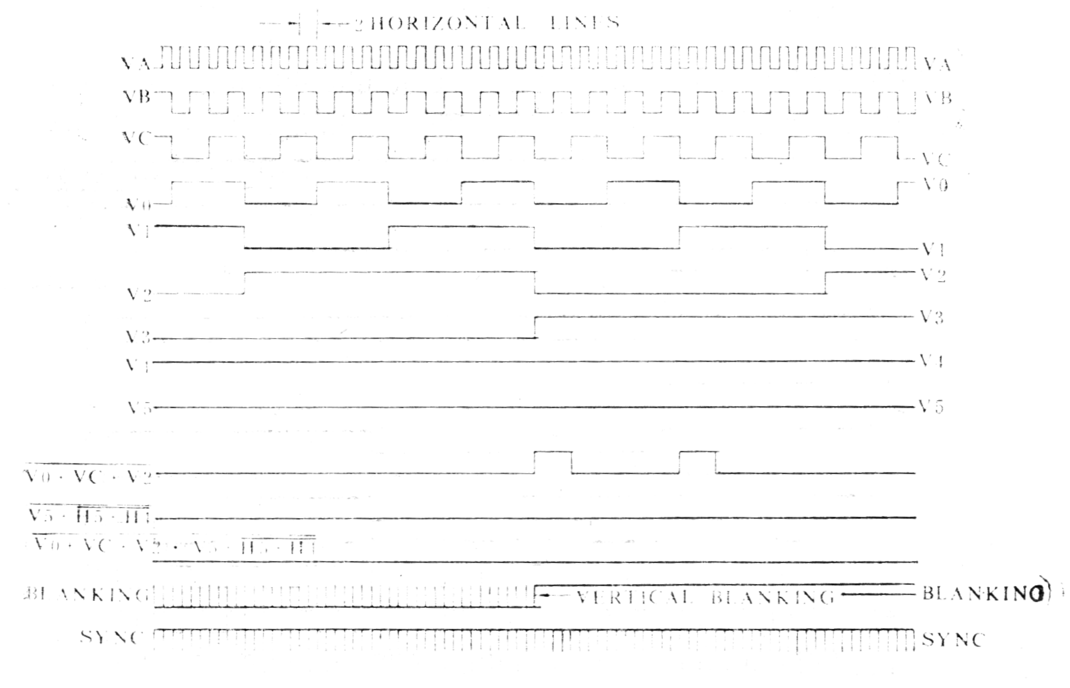


2 HORIZONTAL LINES
VA
VB
VC
V0
V1
V2
V3
V4
V5
$\overline{V0+VC+V2}$
$\overline{V5+\overline{H5}+H4}$
$(\overline{V0+VC+V2})\cdot(\overline{V5+\overline{H5}+H4})$
BLANKING
SYNC

(1)


2 HORIZONTAL LINES
VA
VB
VC
V0
V1
V2
V3
V4
V5
V0+VC+V2
V5+H5+H4
(V0+VC+V2)·(V5+H5+H4)
BLANKING
SYNC

(2)



2 HORIZONTAL LINES
VA
VB
VC
V0
V1
V2
V3
V4
V5
VERTICAL BLANKING
BLANKING
SYNC

(3)

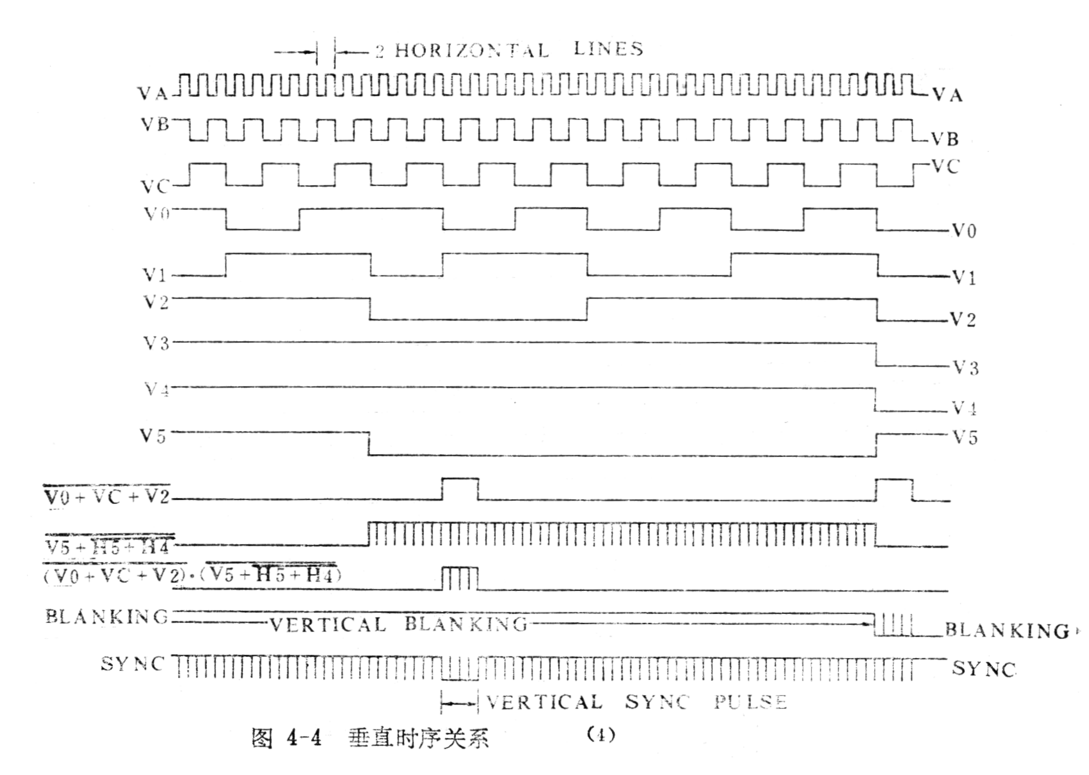

2 HORIZONTAL LINES
VA
VB
VC
V0
V1
V2
V3
V4
V5
$\overline{V0+VC+V2}$
$\overline{V5+\overline{H5}+H4}$
$(\overline{V0+VC+V2})\cdot(\overline{V5+\overline{H5}+H4})$
BLANKING
VERTICAL BLANKING
SYNC
VERTICAL SYNC PULSE

图 4-4 垂直时序关系 (4)

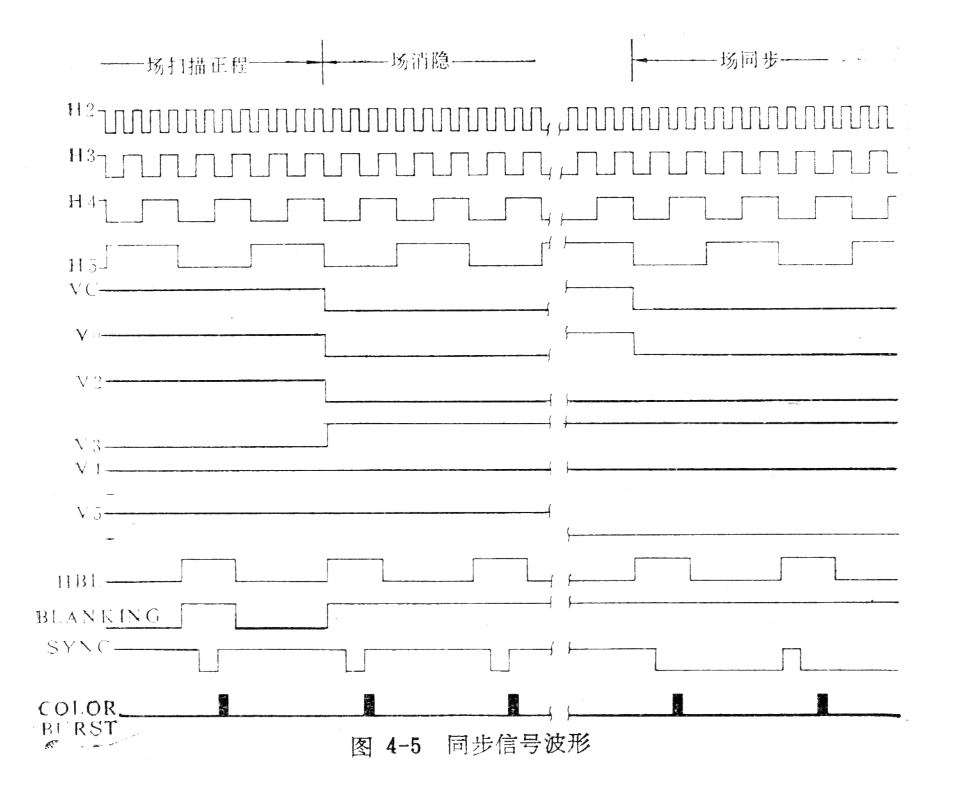


图 4-5 同步信号波形

下面给出这几个信号在 50Hz 场频条件下的逻辑表达式。图 4-5 给出它们的波形。

$$SYNC = \overline{\overline{HBL \cdot H_3} + \overline{V_0 + V_c + V_2 \cdot V_5} + \overline{H_5 + H_4 \times V_3 \cdot V_4}}$$

$$HBL = \overline{H_3 \cdot H_4 + H_5}$$

$$BLANKING = HBL + V_4 \cdot V_3$$

$$COLOR\ BURST = \overline{CREF + H_2} \cdot H_4 \cdot HBL$$

对应 50Hz 场频的情况，GA_1 的几条场频选择管脚应作如下连接：

ACY_2 (GA_1-11) ——不接；

ACY_3 (GA_1-14) ——接 V_5；

ACY_4 (GA_1-15) ——接 V_2。

§ 4.5 视频地址多路转接及地址映射

内存储器4164 RAM，一方面作为 CPU 的存储器，另一方面支持视频显示，其工作是分时的。在 Φ0 期间，CPU 发出的地址分成行列地址送往 RAM，以读写数据，在 Φ1 期间，则由视频地址送往 RAM，以读出数据，送往视频显示部分，在 CRT 上显示字符或图形。因此，在 CPU 地址与视频地址之间，有一个多路转接和相互映射的关系。

地址多路转接器分为两部分。一部分在 GA_4 内部，它完成 RA_0～RA_5 这 6 条 RAM 地址线的多路转接。另外 RA_6 和 RA_7 这两条 RAM 地址线的多路转接器分别在 GA_5 和 GA_7。

多路转接器的控制信号为 Φ0 和 AX。具体地说，当 Φ0 为高时，选择 CPU地址，当 Φ0 为低时，选择视频地

址；当 AX 为高时，选择行地址，当 AX 为低时，选择列地址。

CPU 地址直接送入多路转接器，而视频地址在最后送入多路转接器之前，还要进行一番组合和选择，这是由于下面两点原因造成的：

1. 视频地址线有 14 位，即 H_0、H_1、H_2、H_3、H_4、H_5，VA、VB、VC，V_0、V_1、V_2、V_3、V_4。这 14 位地址的寻址范围为 16384 个单元，但在我们的显示模式中，HIRES 也仅需要 7680 个单元。造成这种浪费的原因是，在消隐期间出现的视频地址是不显示的。为了减少浪费，将 14 位视频地址组合为 13 位，以使视频地址的寻址范围减少为 8192 个单元。

经过组合后的视频地址线为 H_0、H_1、H_2、Σ_0、Σ_1、Σ_2、Σ_3、V_0、V_1、V_2、VA、VB、VC。其中 $\Sigma_0 \sim \Sigma_3$ 由 H_3、H_4、H_5、V_3 及 V_4 组合而来，其表达式如下：

$\Sigma_0 = \overline{V_3} + H_3 + 1$,　　进位 $= C_0$；

$\Sigma_1 = H_4 + \overline{V_4} + C_0$,　　进位 $= C_1$；

$\Sigma_2 = \overline{V_3} + \overline{H_5} + C_1$,　　进位 $= C_2$；

$\Sigma_3 = \overline{H_5} + V_4 + C_2$,　　进位不用。

注意，以上加法是二进制加法，不是逻辑或。

2. 由于整个显示系统占有六个显示缓冲区（文本与低分辨率共同占有两个），必然在不同的显示情况下有不同的地址映射关系。这种不同的映射关系的实现，是依靠屏幕软开关对映频地址进行选择形成的。

通过以上组合和选择而最后形成的视频地址，送入多路转接器，最终形成 RAM 行列地址。

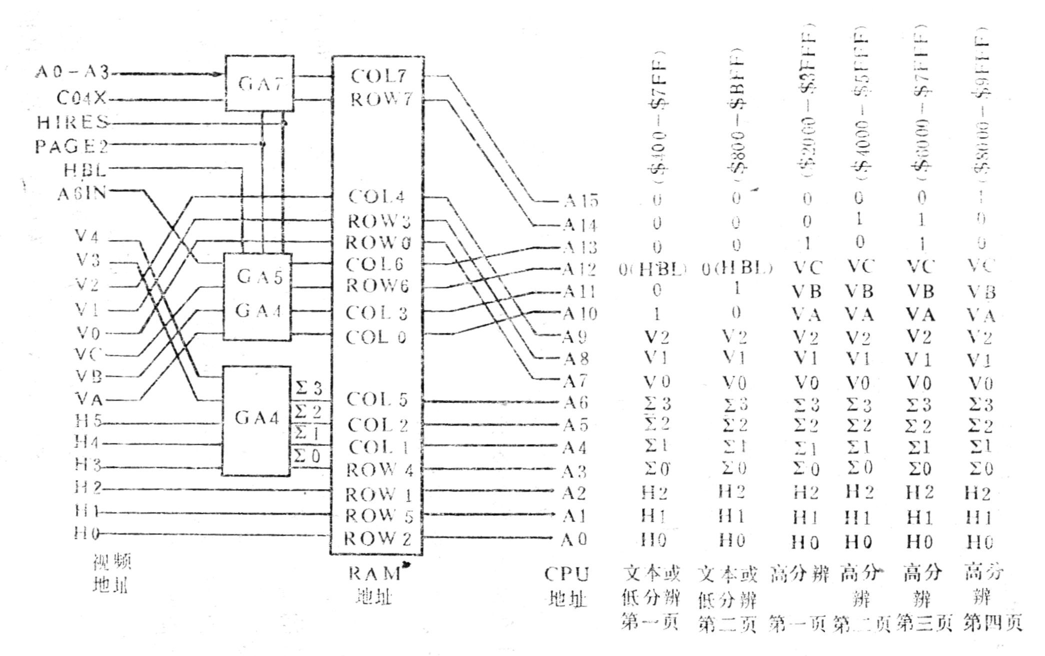


图 4-6 视频地址与 CPU 地址的映射关系

图 4-6 给出了视频地址与 CPU 地址之间的映射关系。图中 TEXT MODE、MIX MODE、与 $PAGE_2$ 的定义见表 4-1。

由这个映射关系即可得到各个图形页的缓冲区地址。下面以 TEXT 模式第一页为例：

左上角：$H_0=0$，$H_1=0$，$H_2=0$，$H_3=1$，$H_4=1$，$H_5=0$，$V_0=0$，$V_1=0$，$V_2=0$，$V_3=0$，$V_4=0$；

（以上关系可由图 4—4 及水平时序图得到）

$\sum 0=V_3+H_3+1=0$，$C_0=1$；

$\sum 1=H_4+V_4+C_0=0$，$C_1=1$；

$\sum 2=V_3+\overline{H_5}+C_1=0$，$C_2=1$；

$\sum 3=H_5+V_4+C_2=0$。

对应于 CPU 地址为 $A_{10}=1$，其余均为 0 。因此屏幕左上角的显示缓冲区地址为 \$400。

§ 4.6 视频数据发生器

存储在 RAM 内的数据，被视频地址读出后，通过 GA_2 锁存并且送到视频数据发生器。发生器根据不同显示模式的要求，将 RAM 数据变换成视频数据（即串行脉冲信号）送往混合器以产生全电视信号。

视频数据发生器的原理图是图 4-7。图中有几根控制线需要说明如下：

GA_{1-22} ——字符显示方式控制线。当 BD_6、BD_7 均为低时，此线为高，即反相显示方式；当 BD_6 为高，BD_7 为低时，此线在高低电平之间来回摆动，即闪烁显示方式，闪烁

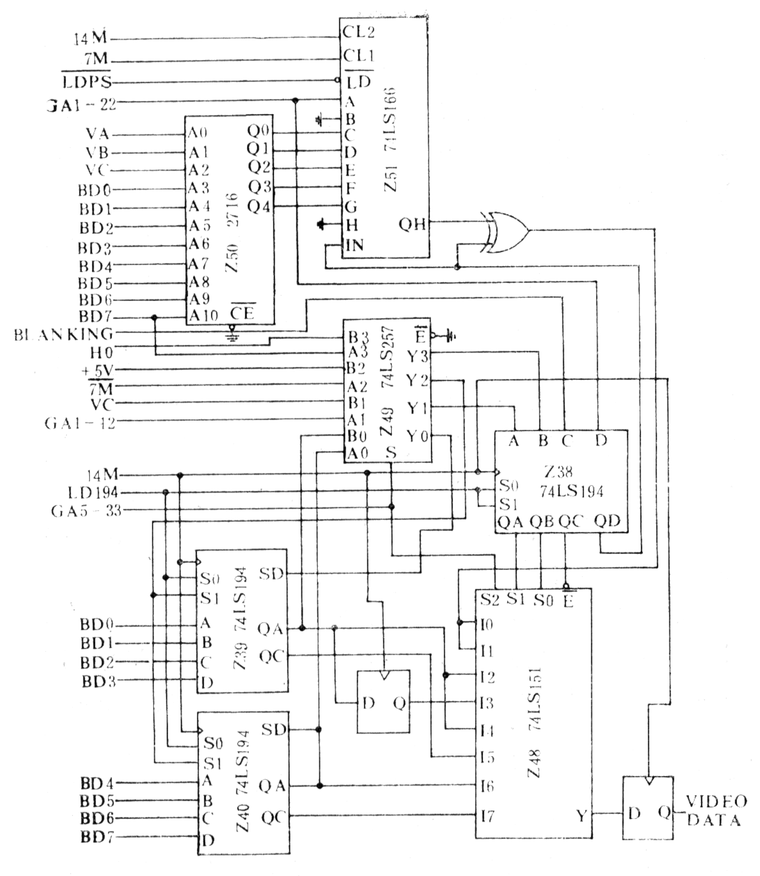


图 4-7 视频数据发生器

频率由 R112 和 C101 决定；在其余状态下，此线均为低电平。

GA1-12 ——文本模式控制线。当显示文本模式或混合

模式的文本部分时，此线为低电平。其逻辑表达式可简化写成：

$$(GA_{1-12})=\overline{\text{TEXT MODE}+\text{MIX MODE}\cdot V_2\cdot V_4}$$

GA_{5-33} ——低分辨率图形模式选择线。当显示低分辨率图形时，此线为高电平，其余情况下皆为低电平，其逻辑表达式可简化为：

$$(GA_{5-33})=\overline{\text{HIRES MODE}+(GA_{1-12})}$$

下面对各种显示模式作一详细说明。

1. 文本模式

文本模式有 $(GA_{1-12})=0$，$(GA_{5-33})=0$。此时 Z_{49} 的 S 端为 0，故 $Y_1=A_1=(GA_{1-12})=0$，$Y_3=A_3=BD_7$。再看 Z_{48}，上述两电平通过 Z_{38} 锁存后加于 S_1与 S_0 端，而其 S_2 端为$(GA_{5-33})=0$，因此整个 Z_{48} 所选择的无非是 I_0 或 I_1，而这两端又都连到异或门的输出，故此时整个电路的结构可简化为图 4-8。

由图可见，$BD_0 \sim BD_7$ 及 VA、VB、VC 共 11 根地址线组成了字符发生器 2716 的地址线。这 11 根地址线的寻址能力是 2K 字节，其中 $BD_0 \sim BD_7$ 用来决定 256 个 ASCII 字符，而 VA～VC 在每个字符的显示过程中变化 8 种状态，组成一个字符的 8 条扫描线。在每寻址到一个点阵后，点阵数据被送到 Z_{51}（74LS166）的输入端，在 $\overline{\text{LDPS}}$、14M 及 7M 三个信号的联合作用下，字符点阵被打入 Z_{51} 并继之移位，最后在 QH 点送出串行波形信号。波形可参看图 4-9。

此串行信号跟（GA_{1-22}）进行异或操作以决定显示方式，最后被 14M 信号所同步，形成视频数据输出。

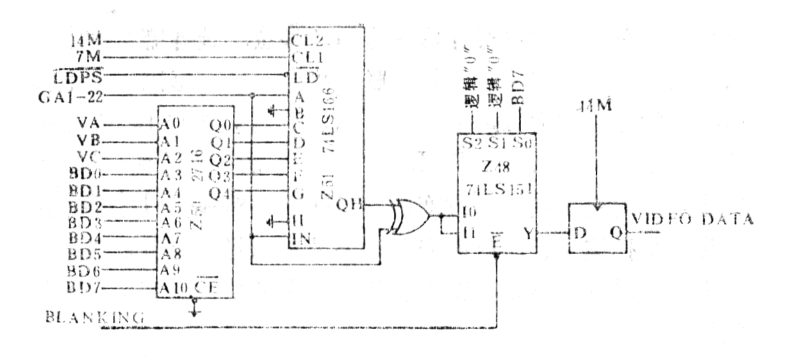


图 4-8 文本模式的视频数据发生器简化原理图

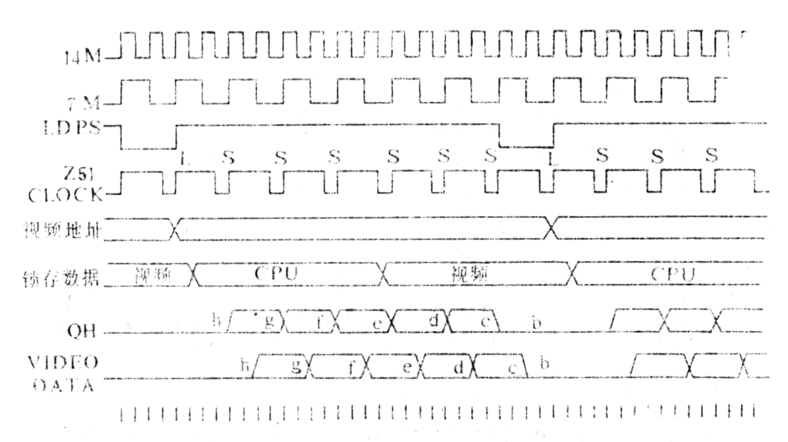


图 4-9 文本模式的时间关系图

2. 低分辨率图形模式

在这个模式下，有（GA1-12）=1，（GA5-33）=1。经过同文本模式相类似的讨论，可得到简化的原理图如图4-10。

由图可见，由 RAM 来的数据被分成两个半字节，每

个半字节在一个四位寄存器中作循环移位，选择哪个半字节的数据送出去由 VC 与 H0 联合决定。H0 每隔一个字符的时间间隔（约为 1μs）变化一次，它可以选择 I4 或 I5 中的一个信号，也可以选择 I6 或 I7 中的一个，而 VC 每扫描 4 行改变一次，它改变的是 D4/D5 对还是 D6/D7对，因此，每扫描 4 行，就改变了半字节的选择。这就形成了每个字节在屏幕上显示成两个色块，每个色块由半字节决定的显示模式。

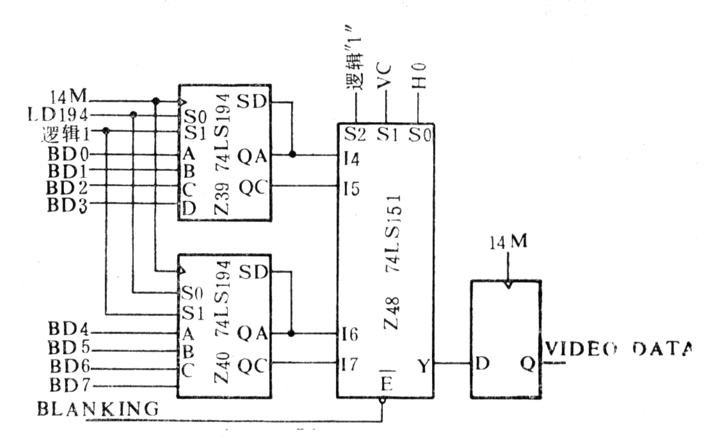


图 4-10 低分辨率图形模式下的视频数据发生器简化原理图

至于每隔一个字符的扫描间隔变化一次 I4/I5，是因为对每个字符的扫描间隔来说，恰好是 3.5 个色同步周期，因此，奇的字符列和偶的字符列之间，其色同步相差 180°。为了在不同的字符列上能显示相同的色彩，需要将输出信号每隔一个字符间隔倒相一次，而移位寄存器 74LS194 的 QA 输出和 QC 输出恰好能满足这个倒相要求。

本模式的时间关系参见图 4-11。

3. 高分辨率图形模式

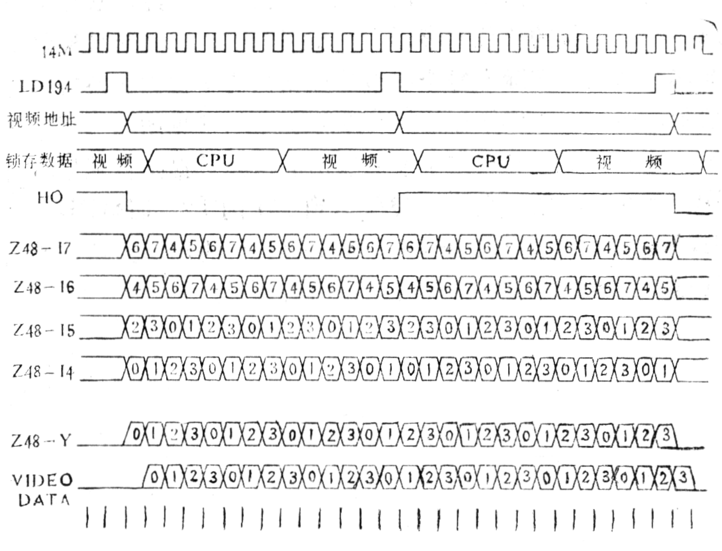


图 4-11 低分辨率模式的时间关系

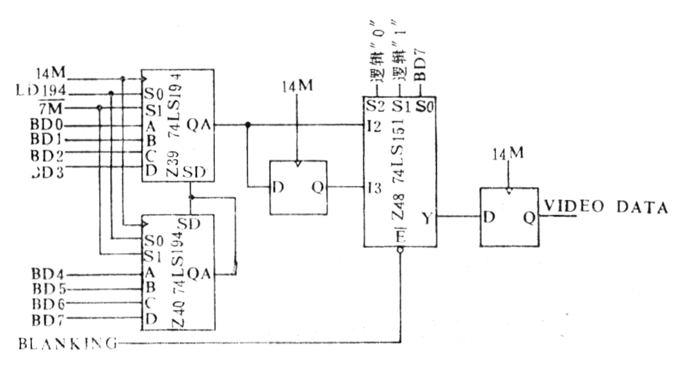


图 4-12 高分辨率图形模式下的视频数据发生器简化原理图

在本模式情况，有 (GA1-12) = 1，(GA5-33) = 0。经过类似的讨论后，可得到简化的原理图如图 4-12。

由图可见，两个 4 位的移位寄存器现在已经串联起来而成为一个 8 位的移位寄存器。由于每两次加载之间只移位了 6 次，因此，实际上每个字节的数据只有 7 位被变换成串行信号，bit 7 并没有出现在串行信号中（参见图 4-13）。但是，这一位数据却决定了最后输出的视频数据是否延迟，当 BD7 = 0 时，数据直接从 QA 送到 151 的 I_2，而当 BD7 = 1 时，数据经过了一段延迟——一个 14M 的周期。正是这点延迟，便改变了该点在屏幕上显示的色彩。

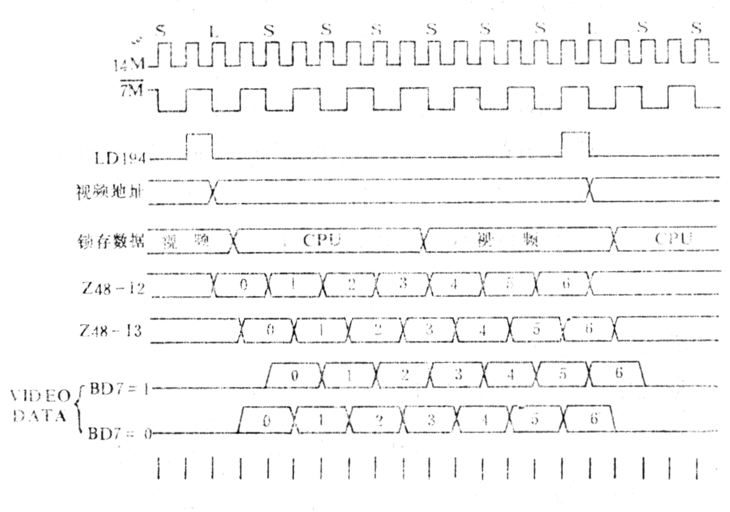


图 4-13 高分辨率图形模式的时间关系

§ 4.7 NTSC-PAL 制式转换器

众所周知，在彩色电视信号中，色彩信息是利用彩色副

载频信号发送的。这个彩色副载信号的相位决定了色彩的色调，其强度决定了色彩的饱和度。为了确定其相位，在全电视信号中加入了一个参考相位，即色同步脉冲。

由于传送彩色副载频信号和色同步脉冲的方式不同，就产生了不同的彩色制式。"小蜜蜂-I" 微型计算机的视频数据发生器产生的信号与同步信号混合后，得到的是 NTSC 制的全电视信号（不包含声音信号），为了适应国内的电视机情况，机内又设置了 NTSC-PAL 制式转换器。下面说明这个制式转换器的工作原理。

1. NTSC 制式与 PAL 制式的不同

NTSC 制式和 PAL 制式的不同，主要有以下几点：

（1）色差信号的带宽不同；

（2）色差信号调制的两个副载频相位不同。NTSC 制为I'信号（123°）和 Q' 信号（33°），PAL 制为U'信号（0°）和 V'信号（90°），且 V' 信号逐行倒相。

（3）色同步信号相位不同。NTSC 制的色同步脉冲相位固定为 180°，而 PAL 制实行逐行倒相，即前一行为 +135° 的 N 行，后一行为 -135° 的 P 行。

（4）除上述 3 点外，由于黑白制式的不同，MONITOR 采用的是 NTSC-M 制，而我国的电视制式为 PAL-D 制，因此彩色副载频的中心频率也不同，前者为3579545±10Hz，后者为 4433619±1Hz。

2. NTSC-PAL 制式转换原理

要完成制式转换，就要考察上述四点不同。其中，第（1）点我们的机器影响不大，因为显示机器图象不必象电视图象那么严格。但后三点则必须解决，否则不能显示彩色。

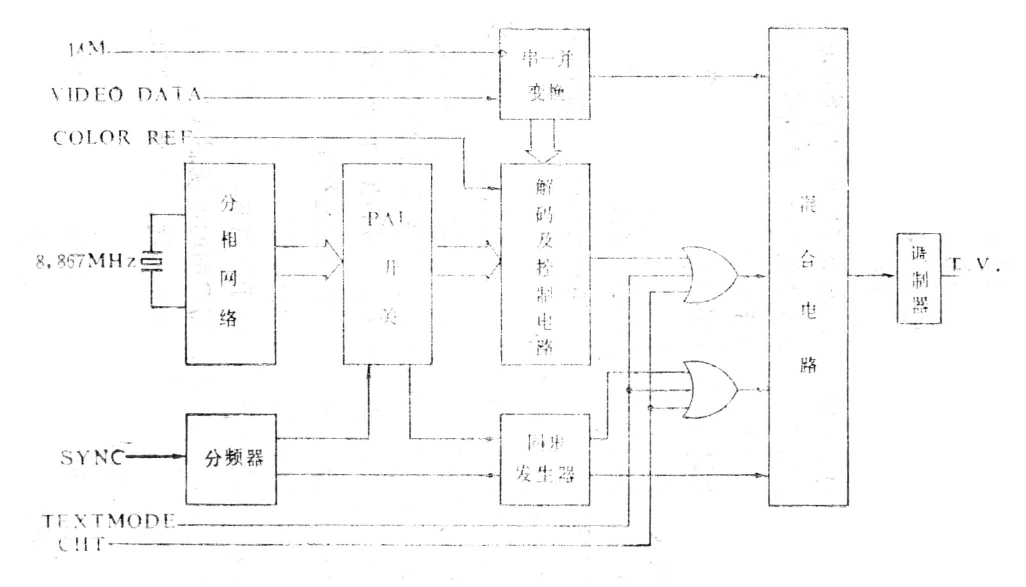


图 4-14 NTSC-PAL 制式转换器原理图

具体的制式转换器原理框图见图 4-14。其中，8.867 MHz 的晶振用来产生 4.434MHz 的彩色副载频及色同步信号，它经过分频后，组合出本机的 16 种不同的彩色信号。行同步脉冲 SYNC 被分频后，用来作为 PAL 开关的控制信号，以实现逐行倒相的目的。NTSC 制的 VIDEO DATA 信号，送入 Z114（74LS164）进行串-并转换，变换后的并行数据送入解码电路，由 COLOR REF 信号进行同步解码后，控制 PAL 制色度信号的输出。

最后输出的四个信号——视频数据、色度信号（GA8-29）、色同步信号（GA8-30）、同步脉冲（GA8-31），被送到由模拟电路组成的混合电路中，最终混合成全电视信号，再送到调制器内，被调制成高频信号送至 T.V 插口。

在色度信号和色同步信号的输出端，接有两个或门，各有两个控制端。其中一个接到 TEXT MODE，这是为了在文本模式显示时，能够关掉彩色，从而使屏幕看起来更清楚些。另一个接到 GA7-12 去，其名称写作 CHT，这是 CHINESE TEXT MODE 的略写。设置这个控制端的本意是要在显示中文状态时（此时必然处于高分辨率图形状态）能够关掉彩色，从而使文本看得更清楚些。不过用户也完全可以在自己认为不需要色彩的时候关掉色度信号，从这一点而言，它更象一个屏幕软开关。它的地址是 $C044/$C045，即 $C044 打开彩色（CHT=0），$C045 关闭彩色（CHT=1）。

整个制式转换器的绝大部分在 GA8 中。当运行系统主盘上一个名为 COLOR DEMOSOFT 的程序时，各主要点的波形见图 4-15。

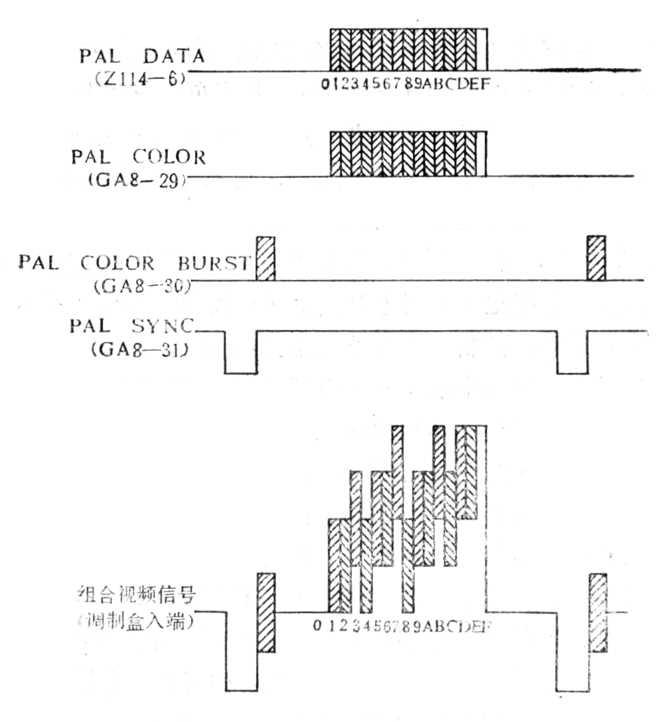


图 4-15 NTSC-PAL 制式转换器各点波形图

第五章　存 储 器

§5.1　概述

在系统 65SC02 CPU 所能直接寻址的 64K 地址空间里存储器占用的地址空间为 $1000～$BFFF，总共60K。其具体分配有以下两种不同情况：

1. 系统在固化的监控程序管理下运行的时候，存储器所占用的地址空间的具体分配是：

地址空间 $0000～$BFFF 被 RAM 所占用。这实际上表示在目前系统所配置的 64K 字节 RAM 中，只有其中的 48K 字节存储单元可以被系统直接寻址。

地址空间 $D000～$FFFF 被 ROM 所占用。这实际上表示系统能够在这 12K 地址空间里对 ROM 直接寻址。系统为了增加在这种情况下的 ROM 的存储容量，还采用了设置软件开关进行体选的方法，把在这种情况下的 ROM 的存储容量扩大到了 32K 字节。

2. 系统在软磁盘操作系统管理下运行的时候，存储器所占用的 60K 地址空间，全部可以被 RAM 所占用（或部分被 ROM 所占用）。此时，系统也可以采用软件开关体选的方法，使系统能够对系统目前配置的 64K 字节的 RAM 寻址。

上述这两种情况下存储器地址空间的具体分配及其有关电路，将在以下有关的小节里详述。

§ 5.2 ROM 存储器电路

1. 27256 EPROM 芯片简介

系统 ROM 存储器是选用了一片 32K 字节的 27256 EPROM 芯片。这个芯片的 28 个引腿定义及功能是:

A14～A0——15 条地址输入引腿。

D0～D7——八条数据输入（编程时输入数据信息）、

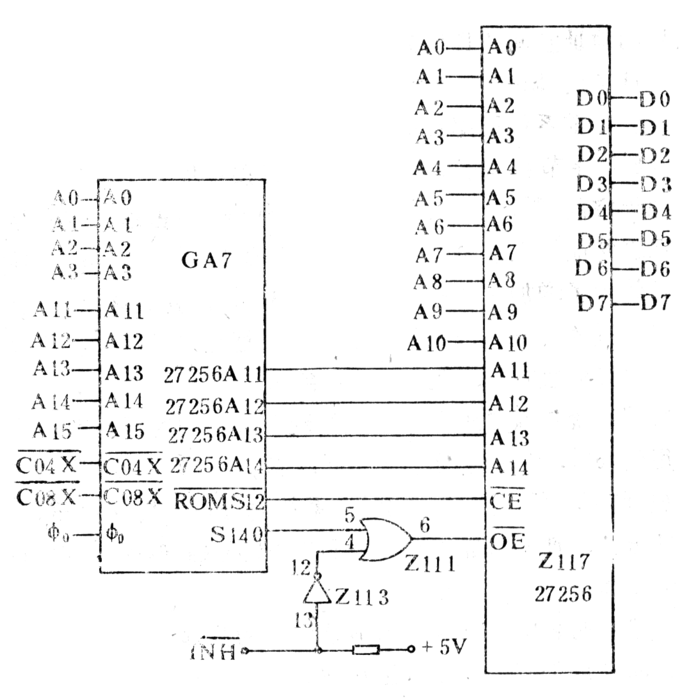


图 5-1 系统 ROM 存储器电路

输出引腿。

$\overline{CE}$ ——片选控制引腿。

$\overline{OE}$ ——允许数据输出引腿。

Vpp——编程电源引腿。

Vcc—— +5V 电源引腿。

GND——地。

2. ROM 存储器电路说明

系统 ROM 存储器电路示于图 5-1。现简要说明其电路原理如下：

(1) 地址输入电路

由于系统对这片 32K 字节的 27256EPROM芯片的直接寻址的地址空间只有 12K，即地址空间 $D000～$FFFF。因此，系统采用了设置软件开关进行体选的方法，以解决对这片 27256 EPROM 芯片的寻址问题。而今，系统是把这片 27256 EPROM 分配在3个存储器体里（0，1 和 2 体）。具体对应的地址分配如图 5-2 所示。

系统规定设置下列4个软件开关来实现对应的体选操作：

C04D：选 0 体，退出 1 体。

C04C：选 1 体，退出 0 体。

C04B：选 0 体，退出 2 体。

C04A：选 2 体，退出 0 体。

为实现上述体选控制功能，现根据图 5-1 所示的电路分别说明一下这个 27256 EPROM 芯片的 15 位输入地址信号的连接情况为：

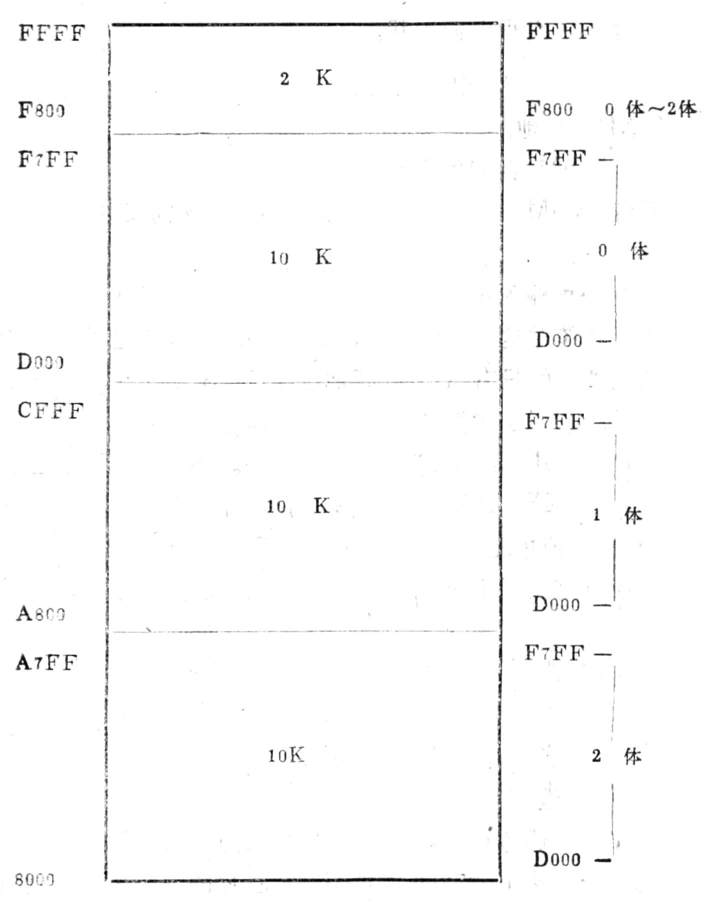


图 5-2 27256 EPROM 芯片地址分配示意图

A. 27256 EPROM 芯片的低 11 位地址输入端 A_{10} ～A_0 是被直接接到系统总线 A_{10}～A_0。它们对这个芯片内部的寻址操作是直接由系统地址完成的。

B. 27256 EPROM 芯片的高 4 位地址输入端 A_{14}～A_{11} 是被接到门阵列芯片的 GA_7 对应的 4 个地址输出端 27256 A_{14}～27256 A_{11}。这 4 个地址信号在系统对 27256 EPROM 芯片寻址时，能够符合下列不同体选情况下各个不同地址空间的要求：

当系统在地址空间 \$F800～\$FFFF 里对 27256 EP-ROM 寻址的时候，GA_7 输出的这 4 个地址信号是一一对应于送到它的 A_{14}～A_{11} 端的系统地址总线的地址信号 A_{14}～A_{11}。这表明地址信号 27256 A_{14}～27256 A_{11} 就是系统地址信号 A_{14}～A_{11}，与系统体选无关，也就是不管系统是选 0 体，1 体还是 2 体，系统总可以在它的地址空间 \$F800～\$FFFF 里对 27256 EPROM 的最高 2K 字节的存储单元直接寻址。

当系统选 27256 EPROM 存储器的 0 体，在地址空间 \$D000～\$F7FF 里寻址的时候，GA_7 输出的 4 个地址信号 27256 A_{14}～27256 A_{11} 也是一一对应于送到它的 A_{14}～A_{11} 端的系统地址总线上的地址信号 A_{14}～A_{11}。系统选 ROM 存储器的 0 体的情况有二：一是设置软件开关 C04D 或 C04B，控制 GA_7 芯片内部电路产生与系统总线地址 A_{14}～A_{11} 一一对应的 27256 A_{14}～27256 A_{11}；二是在系统加电时，GA_7 芯片内部有关电路的起始状态会产生一个有效的软件开关信号 C04D，以控制其产生地址信号 27256 A_{14}～27256 A_{11}。根据图 5-2 所示的地址分配示

意图，系统就可以在 27256 EPROM 芯片的内部地址分布的 $D000～$F7FF 里寻址。读 27256 EPROM 芯片对应的地址空间存储单元的内容。

当系统选 27256 EPROM 存储器的 1 体，在地址空间 $D000～$F7FF 里寻址的时候，GA7 芯片输出的 4 个地址信号 27256 A14～27256 A11 与送到它的 A14～A11 输入端上系统地址总线上的地址信号 A14～A11 是不对应的。这时，系统设置的软件开关信号 C04C 将由系统送到这个 GA7 芯片的 C04X 信号和系统地址信号 A3～A0，在 GA7 芯片内部的体选控制电路把从其地址输入端 A14～A13 送来的对应于地址空间 $D000～$F7FF 的地址信号转换成对应于在 27256 EPROM 芯片中寻址地址空间 $A800～SCFFF 的地址信号 27256 A14～27256 A11。读这个 27256 EPROM 芯片内地址空间的存储单元的内容。在这种情况下，系统只能用设置软件开关 C04D 的办法退出存储器 1 体，进入存储器 0 体。而不能直接从存储器 1 体转到存储器 2 体。

同样，当系统选 27256 EPROM 存储器的 2 体，在地址空间 $D000～$F7FF 里寻址的时候，系统则应设置软件开关信号 C04A，它在 GA7 芯片里控制其内部体选控制电路，把从其地址输入端 A14～A11 送来的对应于地址空间 $D000～$F7FF 的地址信号转换成对应于在 27256 EPROM 芯片内地址空间的存储单元内容。在这种情况下，在它退出 2 体时，也只能进入存储器 0 体，不能直接转到存储器 1 体。

（2）片选（$\overline{CE}$）控制电路

27256 EPROM 芯片的片选端 $\overline{CE}$ 是由 GA7 芯片所产生的 $\overline{ROMS_{12}}$ 控制信号控制的。只要系统在地址空间 \$D000～\$F7FF 里寻址，由系统地址总线的高 4 位送到 GA7 芯片的地址信号 A_{15}～A_{12}，将在 GA7 里有关的译码电路译码，且在系统时钟信号 Φ0 为高电平期间，产生这个低电平有效的 $\overline{ROMS_{12}}$ 信号，以实现对 27256 EPROM 芯片的片选控制功能。

（3）允许数据输出 $\overline{OE}$ 控制电路

27256 EPROM 芯片的允许数据输出端 OE 是由 GA7 芯片所产生的 S_{140} 控制信号控制的。只要系统是在地址空间 \$D000～\$F7FF 里对这个 27256 EPROM 芯片寻址，而不是对系统扩展的 16K 字节 RAM 寻址，由系统地址总线的高 4 位送到 GA7 芯片的地址信号 A_{15}～A_{12}，将在 GA7 里有关的译码电路译码，从 GA7 芯片的 S_{140} 端输出一个相应的无效低电平信号，送到或门 Z_{111} 的引腿 5 输入端。这时，如果从扩展槽上送来的请求信号 INH 为无效高电平，那么或门 Z_{111} 的引腿 6 输出端就会输出一个相应的低电平信号，送到 27256 EPROM 的 OE 端，允许其 D_7～D_0 端输出被读的数据信息。反之，如果系统是地址空间 \$D000～\$F7FFF 里对系统扩展的 16K 字节 RAM 寻址或是从扩展槽上送来的请求信号 INH 为有效低电平的话，那么都会使或门 Z_{111} 的引腿 6 输出端输出高电平信号控制 27256 EPROM 芯片的 OE 端，禁止其 D_7～D_0 端输出任何数据信息。

（4）数据输入、输出电路

27256 EPROM 芯片的 8 位数据端 D_0～D_7 是被直接

接到数据总线的 $D_7 \sim D_0$ 上。在系统以任何方式读这个 27256 EPROM 芯片时，其数据信息将输出到系统数据总线。只在对这个芯片编程时，数据信息才会被输入到这个芯片里。

§ 5.3 RAM 存储器电路

1. 4164 RAM 芯片简介

系统 RAM 存储器选用的是 64K×1 位的动态随机存储器。这个芯片的 16 条引腿定义及功能是：

$A_7 \sim A_0$——8 条地址输入引腿

D——数据输入引腿

Q——数据输出引腿

$\overline{RAS}$ ——8 位行地址输入选通控制引腿

$\overline{CAS}$ ——8 位列地址输入选通控制引腿

$R/\overline{W}$——数据读（高电平）/写（低电平）控制引腿

V_{CC}——+5V 电源

GND——地

2. 行、列地址转换电路

每一个系统 RAM 的 64K×1 位芯片在寻址时所需的 16 位地址信号是分时用一个 8 位的行地址信号和一个 8 位的列地址信号送进去的。如图 5-3 所示，分别由 GA_7 芯片和 GA_4 芯片产生的行/列地址信号 $RA_7 \sim RA_0$ 被一一对应地接到 8 个 4164RAM 芯片的相应的地址输入端 $A_7 \sim A_0$。下面分别说明一下系统地址总线上的 16 位地址信号 $A_{15} \sim A_0$ 转换成 8 位行/列地址信号 $RA_7 \sim RA_0$ 的有关电路（视频地址转换或 8 位行列地址信号 $RA_7 \sim RA_0$

的有关电路在第四章里已有说明）。

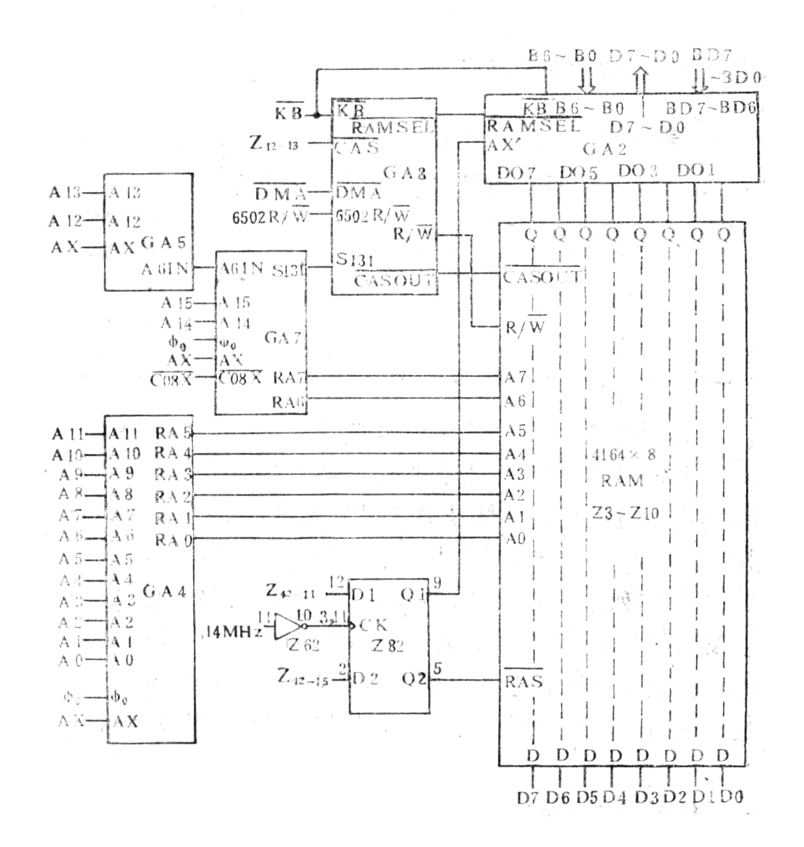


图 5-3 系统 RAM 存储器电路

(1) RA_5～RA_0 的转换电路

GA_4 门阵芯片把系统地址总线上的 12 个低位地址信号 A_{11}～A_0，经过系统时钟信号 Φ0 和地址选择信号 AX

译码选通之后，由这个芯片内部的有关电路产生 6 个与上述 12 位地址信号对应的低位行/列地址信号 RA_5～RA_0。其对应的转换关系如表 5-1 所示。

表 5-1　系统总线地址信号 A15～A0 与行/列地址信号 RA7～RA0 的转换关系

	RA0	RA1	RA2	RA3	RA4	RA5	RA6	RA7
行地址	A1	A3	A8	A0	A2	A7	A12	A14
列地址	A6	A9	A11	A5	A4	A10	A13	A15

（2）RA_6，RA_7 的转换电路

GA_5 门阵列芯片把系统地址总线上的两个信号 A_{13} 和 A_{12}，经系统地址选择信号 AX 选通之后，由这个芯片内部的有关电路产生一个与行/列地址有关的信号 A_6IN。它被送到 GA_7 门阵列芯片的相应输入端之后，又经系统时钟信号 Φ0 以及系统在地址空间 \$D000～\$FFFF 里对 RAM 寻址时，在这个 GA_7 芯片里所产生的有关的译码控制信号选通，最后产生行/列地址信号 RA_6 。其对应关系亦示于图 5-1。

GA_7 门阵列芯片把系统地址总线上的两个地址信号 A_{14} 和 A_{15}，经系统地址选择信号 AX 和系统时钟信号 Φ0 译码选通之后，由这个芯片的内部有关电路产生对应的行/列地址信号 RA_7。其对应的转换关系亦如表 5-1 所示。

3．行、列地址选通控制信号 $\overline{RAS}$ 、$\overline{CASOUT}$

（1）行地址选通控制信号 $\overline{RAS}$

由系统时序电路中的 Z42 的引腿 15 输出端产生的低电平有效的行选控制信号，再经过图 5-3 的一个 D-触发器 (Z32) 延迟大约 35ns 之后，由其 Q2 输出端输出的就是送到所有 8 个 RAM 芯片的行地址选通输入端 $\overline{RAS}$ 的行地址选通控制信号 RAS。

（2）列地址选通控制信号 $\overline{CASOUT}$

当系统在地址空间 $D000～$BFFF 里直接对 RAM 芯片寻址的时候，或者当系统在地址空间 $D000～$FFFF 里对扩展的 16K 字节 RAM 体选寻址的时候，GA7 门阵列总会产生一个对 RAM 寻址的低电平有效的控制信号 S131。它被送到 GA8 门阵列芯片对应的输入端，以便把从系统时序电路中 Z42 的引腿 13 输出端产生的低电平有效列选控制信号 CAS，经 S131 信号选通之后产生送到所有 8 个 RAM 芯片列地址选通输入端 CAS 的列地址选通控制信号 $\overline{CASOUT}$。

4. 扩展 RAM 的地址影射

由于系统随机存储器选用的是 64K×1 位的 4164 芯片，8 片这样的 RAM 芯片可以提供 64K 字节存储单元。除去系统已将其 48K 字节存储单元分配占用系统的存储器地址空间 $0000～$BFFF 之外，尚有 16K 字节存储单元（在这个 8 片 4164 RAM 芯片中的物理地址为 $C000～$FFFF)可以作为系统的扩展 RAM。目前，在这个系统里是用来规定这 16K 字节存储单元设置不同的体选软件开关的办法，以控制它分别占用系统地址空间 $D000～$FFFF 的 12K 和地址空间 $D000～$DFFF 的 4 K

（参看图 5-4）。在系统扩展 RAM 的时候，系统必然是在软磁盘操作系统管理下运行。这时，系统对原在地址空间 \$D000～\$FFFF 里的 ROM 的寻址功能应全部被禁止（在系统寻址 RAM 中物理地址空间为 \$D000～\$FFFF 12K 字节存储单元时）或仍然允许（在系统寻址 RAM 中物理地址空间为 \$D000～\$DFFF 4K 字节存储单元时），目的是避免在系统寻址时（尤其是读操作寻址）上述地址空间时发生冲突。

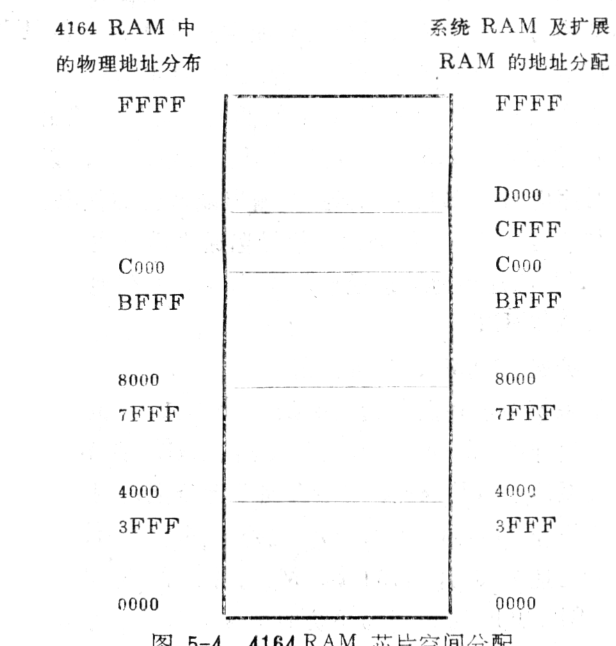


图 5-4　4164 RAM 芯片空间分配

根据以上分析，为满足扩展 RAM 地址影射的要求，GA7 门阵列中的部分电路为设置不同的软件开关，把由系

统地址译码控制电路中的 Z116 引腿 15 输出端送来的地址译码控制信号 C08X 和系统地址信号 A3、A1 和 A0 进行锁存和译码，以产生相应的控制信号，最后控制产生对应于不同的体选软件开关所需的控制信号 S140 和 S131。实现对系统扩展 RAM 的地址影射以及在必要时禁止对 ROM 的读操作。系统所应设置的软件开关的功能与送到 GA7 芯片的系统地址信号 A3、A1 和 A0 的关系如下：

A3＝1时（对应于软件开关为 C088～C08F），系统 4164 RAM 中的物理地址空间为 $C000～$CFFF 的 4K 字节存储单元被影射到系统地址空间 $D000～$DFFF 里。

A3＝0时（对应于软件开关为 C08X～C087），系统 4164 RAM 中的物理地址空间为 $D000～$FFFF 的 12K 字节存储单元被直接影射到系统地址空间 $D000～$DFFF 里。

A1 和 A0 的代码值将决定上述软件开关中系统对扩展 RAM 和系统ROM（在系统地址空间 $D000～$FFFF 里）的读、写操作允许或禁止的功能。其对应关系如表 5-2 所示：

表 5-2 地址信号 A1 和 A0 功能表

A1	A0	软件开关功能
0	0	允许读 RAM，禁止读 ROM
0	1	允许写 RAM，允许读 ROM
1	0	禁止读 RAM，允许读 ROM
1	1	允许读/写 RAM，禁止读 ROM

据此，可以根据系统对扩展 RAM 功能的不同要求，组合上列软件开关来实现所需的体选操作。

5. $R/\overline{W}$ 控制信号

来自 65SC02 CPU 芯片的 $R/\overline{W}$ 端的读（高电平有效）、写（低电平有效）控制信号 6502 $R/\overline{W}$，在系统 DMA 请求信号 $\overline{DMA}$（低电平有效）为无效高电平时，经 GA_8 芯片选通之后，产生一个读/写控制信号 $R/\overline{W}$（参看图 5-3），送到所有 8 个 RAM 芯片的读/写控制端 $R/\overline{W}$，以实现对这些系统 RAM 芯片的读/写控制。

6. 数据信息输入

8 个系统 RAM 芯片的 8 个数据信息输入端D是被一一对应地接到系统数据总线 $D_7 \sim D_0$ 上。系统数据总线上的 8 位数据信息可以直接写进这 8 个 RAM 芯片的对应的存储单元。

7. 数据信息输出

8 个系统 RAM 芯片的 8 个数据信息输出端 Q 是被一一对应地接到 GA_2 门阵列芯片的数据引腿 $DO_7 \sim DO_0$。在系统时序电路中 Z_{42} 的引腿 11 输出端（$\overline{Q_3}$ 端）输出的地址选通信号，经 Z_{82} 的另一个触发器被延迟 35ns之后（参看图 5-3），这个触发器的引腿 9 输出端（Q1 端）输出的地址选通信号 AX′，将被送到 GA_2 芯片的 AX′ 输入端。这个信号的上升沿会把上述的从 8 个 RAM 芯片送到 GA_2 芯片的 $DO_7 \sim DO_0$ 端的 8 位数据信息锁存到这个 GA_2 里。这时，如果从 GA_7 芯片送出来的 S131 信号为低电平（即系统正在对系统 RAM 寻址），GA_8 芯片的 $\overline{RAMSEL}$ 输出端会相应地产生一个低电平有效的控制信

号，送到 GA_2 芯片的 RAMSEL 输入端，以控制 GA_2 芯片把从 $DO_7 \sim DO_0$ 端送进来的 8 位读 RAM 芯片的数据信息送到系统数据总线 $D_7 \sim D_0$（同样，在系统接收键盘输入数据信号时，也是由键盘送来的控制信号 $\overline{KB}$，使GA_8 芯片产生控制信号 $\overline{RAMSEL}$，控制 GA_2 芯片把从键盘送来的数据信息 $B_6 \sim B_0$ 送到系统数据总线 $D_7 \sim D_0$）。

另外，在从系统 8 个 RAM 芯片读出屏幕显示的数据的时候，由于在视频显示周期，GA_7 芯片给出的控制信号 S_{131} 是高电平输出，从而使 GA_2 芯片的 $\overline{RAMSEL}$ 端被 GA_8 给出的高电平有效的 $\overline{RAMSEL}$ 信号所控制，把这时从 RAM 芯片读出的 8 位数据信息经 GA_2 芯片的屏幕显示数据输出端 $BD_7 \sim BD_0$，送到系统字符发生器等有关电路。

第六章　外围设备及接口

§6.1　概述

XMF-I 可连接各种外围设备。某些基本的外围设备的接口已在主机上设置好了，这将给用户带来方便并减少用户的额外开销。我们称这些接口为“主板上的 I/O”，如键盘、打印机接口等。为了方便用户扩展各种外围设备，XMF-I 还可配接一个扩展箱，可用电缆与主机连接。扩展箱内备有电源、两个软盘驱动器、一个驱动器接口卡和五个 50 芯扩展插座。扩展 I/O 所需的系统时钟、总线、I/O 地址选通信号及电源等都已按照 Apple-Ⅱ 的引脚定义连接在插座上。

主机上的 I/O 包括：

键盘接口——可用扁平电缆连接键盘板；

打印机接口——主机后侧有一个连接插座；

CRT 接口——主机后侧有一个轴型单芯插座输出；

PAL 制式彩色电视视频信号接口——主机后侧有一个轴型单芯插座输出；

语言卡接口——系统内已将 RAM 扩展到 64K；

汉卡接口——汉字处理硬件、软件均已在主机内设置；

录音机外存储器接口——主机后侧有输入、输出轴型插座各一个；

游戏棒接口——主机后侧有 D 型 9 芯连接插座；

扬声器发声口——已设置于主机板上；

在扩展箱中，设有二个软盘驱动器和接口卡。

XMF-I 采用存储器统一编址方式。考虑到与 Apple-II 的兼容，两者 I/O 地址是一致的。XMF-I 把 $C000～$CFFF 这 4K 地址空间用作 I/O 专用区。其中 $C000～$C07F 用作主板上 I/O 接口地址，其余地址用于扩展

表 6-1 I/O 口地址及功能

功　能	十进制地址	十六进制地址	访问方式(R/W)
键盘输入	49152(-16384)	$C000	R
清键盘选通	49168(-16368)	$C010	R
扬 声 器	49200(-16336)	$C030	R
盒带输出	49184(-16352)	$C020	R
盒带输入	49248(-16288)	$C060	R
开关量输出（输出"0"的地址，地址加1则输出"1"）	49240(-16296) 49242(-16294) 49244(-16292) 49246(-16290)	$C058 $C05A $C05C $C05E	R/W ☆ R/W ☆ R/W ☆ R/W ☆
开关量输入	49249(-16287) 49250(-16286) 49251(-16285)	$C061 $C062 $C063	R R R
摸拟输入 摸拟复位	49252(-16284) 49253(-16283) 49254(-16282) 49225(-16281) 49264(-16272)	$C064 $C065 $C066 $C067 $C070	R R R ☆ R ☆ R/W
实用选通 打印机数据 打印机"忙"	49216(-16320) 49296(-16240) 49536(-16000)	$C040 $C090 $C180	R W R

注：打☆的接口 I 型上暂未设置。

I/O 接口。

主机上的 I/O 地址分配如表 6-1。

表 6-1 中开关量输入和摸拟输入用于游戏棒控制。摸拟输入端能用于检测 150K 左右电位器阻值的变化。

由于电路的特殊接法，上表中的 I/O 只有键盘、打印机接口是八位标准并行接口。其它接口有的仅输入一位，有的仅使用地址线。

$C000～$C07F 之间的地址有一些被用作系统控制软开关。

扩展 I/O 地址的分配是：

$C080～$C0FF 分为八组，每组 16 个地址，引到各扩展插座上，由八个 $\overline{\text{DEVSEL}}$ 选通。$C100～$C7FF 分成 7 段，接到扩展 1#～7# 插座，作为各个扩展 I/O 专用 ROM 地址。由各个 $\overline{\text{I/O SEL}}$ 信号选通。

$C800～$CFFF 作为扩展 I/O 共用 ROM 地址区，共 2K，由 $\overline{\text{I/O STB}}$ 选通。

§ 6.2 I/O 地址译码

图 6-1 是 I/O 地址译码原理图。图中，三片 3-8 译码器 74LS138 以外的门电路设计在门阵列中。图中的输出信号 $\overline{\text{C00X}}$～$\overline{\text{C07X}}$ 八个地址选通信号用于主板上的 I/O 电路，$\overline{\text{DEVSEL0}}$ ～$\overline{\text{DEVSEL7}}$ 等八个选通信号、$\overline{\text{I/O SEL1}}$～$\overline{\text{I/O SEL7}}$ 等七个选通信号以及 $\overline{\text{I/O STB}}$用于扩展 I/O 接口。

$\overline{\text{I/O STB}}$ 有效的条件是 A15 = A14 = 1，A13 = A12 = 0，A11 = 1，也就是说，CPU 访问 $C800～$CFFF 地址

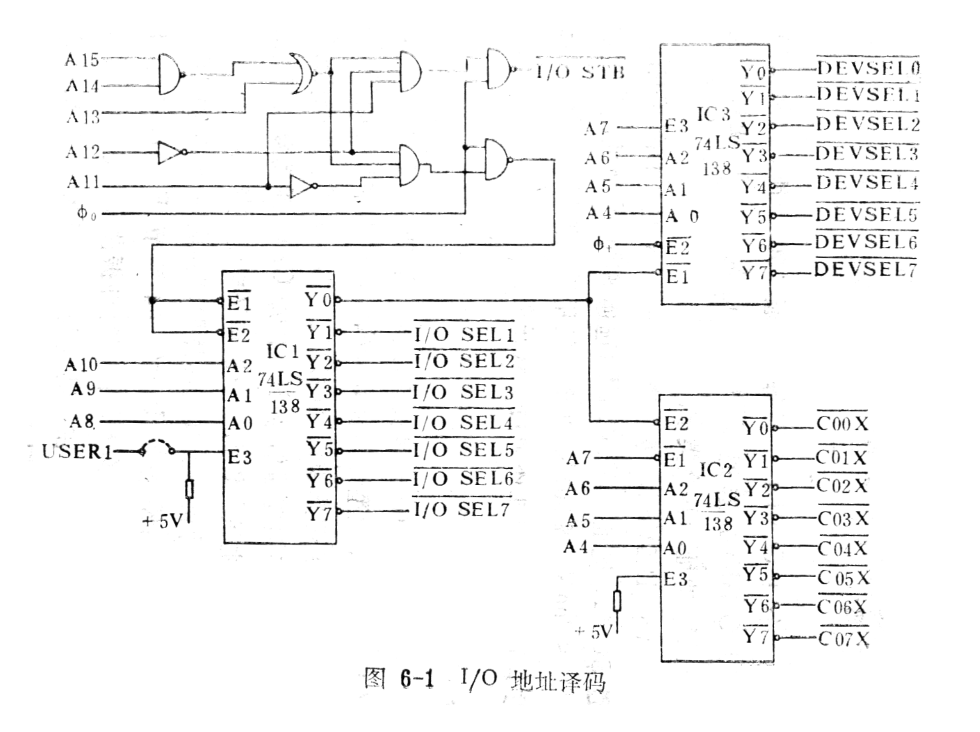

A15
A14
A13
A12
A11
ϕ_0
I/O STB
IC3
74LS
138
A7
A6
A5
A4
ϕ_1
DEVSEL0
DEVSEL1
DEVSEL2
DEVSEL3
DEVSEL4
DEVSEL5
DEVSEL6
DEVSEL7
IC1
74LS
138
A10
A9
A8
USER1
+5V
I/O SEL1
I/O SEL2
I/O SEL3
I/O SEL4
I/O SEL5
I/O SEL6
I/O SEL7
IC2
74LS
138
C00X
C01X
C02X
C03X
C04X
C05X
C06X
C07X

图 6-1 I/O 地址译码

时，这个信号有效。这就是扩展 I/O 共用 2K ROM 地址的选通信号。Φ0 参加译码保证了 CPU 访问存储器周期时该信号才有效。

分配给 I/O 4K 地址中的前 2K \$C000～\$C7FF 译码成三部分：$\overline{\text{I/O SEL}}$ 、$\overline{\text{DEVSEL}}$ 和主机板上 I/O 地址。

七个 $\overline{\text{I/O SEL}}$ 信号每个选通 256 个地址，从图中可知，$\overline{\text{I/O SEL}_1}$ 选通 \$C1XX 地址，$\overline{\text{I/O SEL}_2}$ 选通 \$C2XX 地址，其余类推。

八个 $\overline{\text{DEVSEL}}$每个选通 16 个地址，$\overline{\text{DEVSEL}_0}$ 选通 \$C08X 地址，$\overline{\text{DEVSEL}_1}$选通 \$C09X 地址，其余类推。

\$C00X～\$C07X 八个地址用于主板 I/O，还要进一步译码。部分地址还被用于软开关。

§ 6.3 键盘和接口

XMF-I 使用一单板式键盘。键盘上有 53 个键。板上用一片单片微型计算机片 8048 作键扫描和键处理。一片 74LS74 用于键选通信号的发生。键盘除了向主机传送按键 ASCII 码和键选通信号外，主机 CPU 手动复位键也装在键盘板上。主机与键盘用一根扁平电缆连接。

8048 单片机不加其它芯片即可构成一个完整的系统。8048 内含有 1K 掩膜 ROM，64 字节 RAM，三个 8 位 I/O 口，三个测试输入，内部时钟振荡器，这些对于用在键盘电路上很合适。在本键盘电路中，I/O 口 PA_0～PA_7、PB_0～PB_7 用于键扫描（电路如图 6-2），CONTROL 和RESET 两键不在键扫描矩阵上。DB_0～DB_7 则用于向

主机传送 ASCII 码，其中 DB7 用于产生键选通信号。

从图 6-2b 可知，8048 的 2、3 脚上接有一个晶体和二个小电容，和内部振荡电路一起构成时钟电路。通过 4 脚上的上电复位，进入内部 ROM 中的工作程序。工作程序通过 2 个 8 位口扫描键盘。查找被按下的键，并产生相应的 ASCII 码，送到数据口 D0～D6，同时 D7 送一个“1”，这使 74LS74 右边的一个触发器 Q＝1（如果CLR＝1），这个“1”通过插座 6 脚送到主机板，作为按键标志。 74LS74 左边一个触发器的时钟是 8048 的内部时钟，D 端接在 $C010（CLR STR），平时为“1”。当主机从键盘获得一个按键 ASCII 码后，它就由 $C010发一个 0.5us 的负脉冲，这使左边触发器 Q＝0（RESET无效），使右边的 D触发器被清除，也就消除了按键标志。如果再按键，将重演上述过程。

当键盘上 RESET 和 CONTROL 两键同时按下时，将产生主机 CPU 硬件复位。也使按键标志复位，但 8048 不复位。

CONTROL 键按下时，一个低电平输入 8048 T2 测试端，软件将使同时按下的一个键产生第二种键值。

主板上的键盘输入接口由二片带使能端的四二选一开关 74LS257 构成。在 RAMSEL 有效时，当读 $C000 时，键盘送来的 7 位 ASCII 码和按键标志被接上数据总线，不访问这个地址而读 RAM 时，各 RAM 的读出端被接上数据总线。

§ 6.4 收录机外存储器

XMF-I 的收录机外存储器的软硬件设计保证了使用

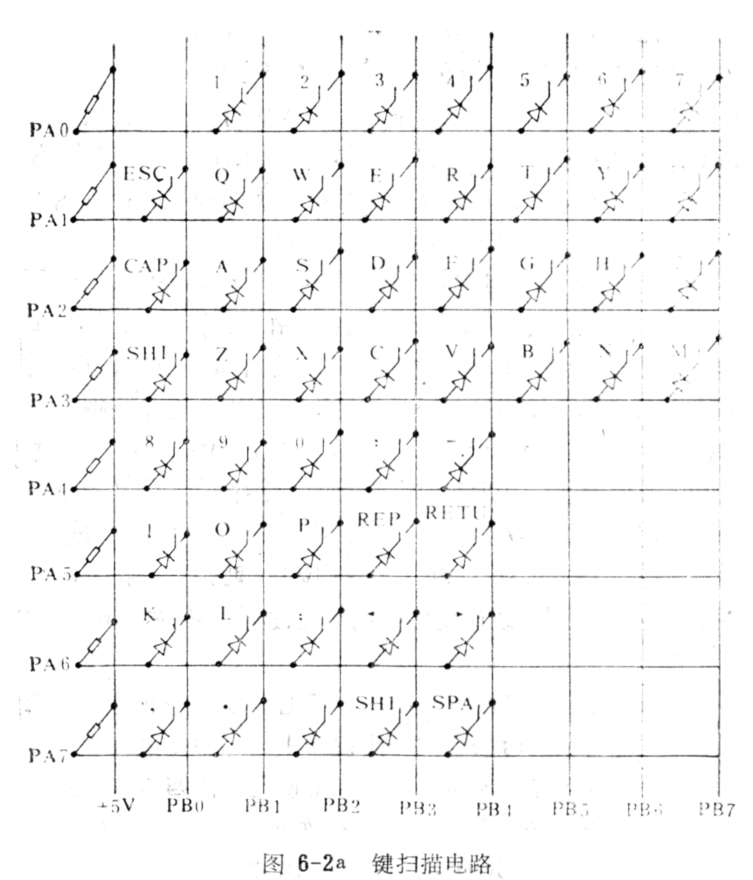


图 6-2a 键扫描电路

一般家庭用收录机也能高可靠地工作。作为本机特色之一，录音机可以带文件名存取。在从录音机读入文件时，带文件名可以使输入相当方便。计算机会自动寻找同命令相同的文件，丢弃文件名不一致的文件而只在屏幕上给出已发现的文件的名称，并继续寻找下一个文件。

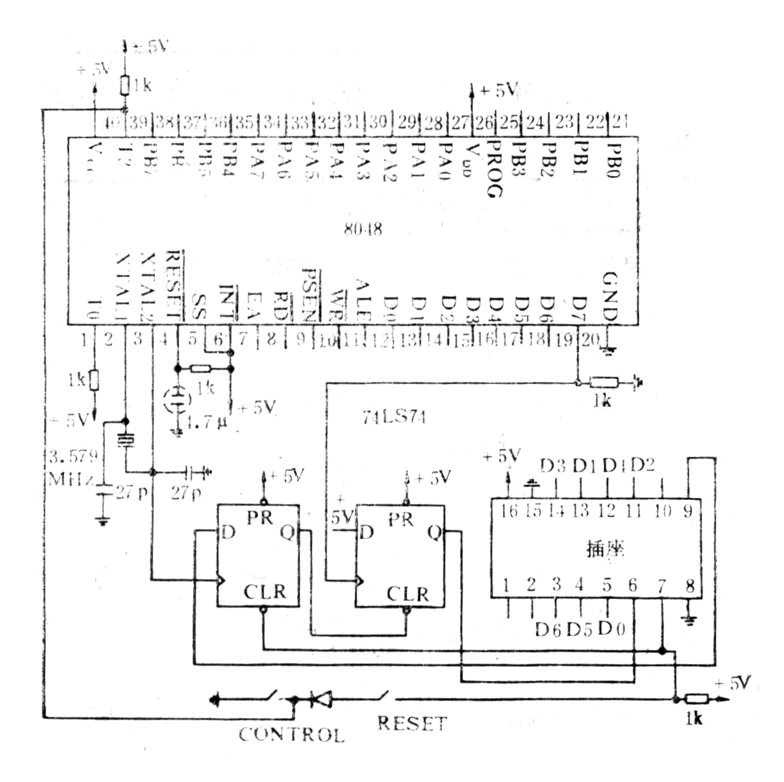


图 6-2b 键盘电路

我们称记录在磁带上的数据为“记录”。每个记录包括 10 秒钟左右的起始段，接着是同步位和实际的数据和校验和。

在监控状态下，二进制文件和机器码作为一个记录进行处理。在 BASIC 状态下，BASIC 文件和二进制文件均被记为二个记录，第一个是文件长度，后一个是程序本身。就是说，二进制文件（监控状态下）的记录格式为：

起始段	同步位	数　据	检验和

BASIC 文件和 BASIC 状态下的二进制文件的记录格式为：

起始段	同步位	数　据	检验和	起始段	同步位	数　据	检验和

起始段的引导信号、同步位、数据的“0”和“1”用不同的脉冲宽度来表示：

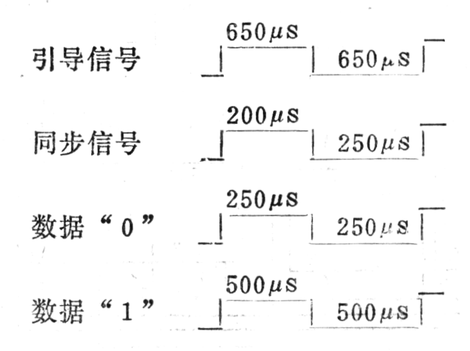


图 6-3 是收录机接口电路。在上部盒带输出电路中，来自主板上 I/O 译码电路的 $C02X 选通信号作为一个 D 触发器的时钟，D 触发器接成二分频电路。在进行盒带存储时，用软件定时产生 $C02X 选通信号，在 $\overline{Q}$ 端就可以获得不同宽度的脉冲，作为引导信号、同步位、数据“0”和“1”。通过电阻衰减到约 30mV，由一个插座输出。这个信号可输入录音机的 MIC 进行录制。

录音机到计算机的接口电路包括一个整形电路和一个 8 选 1 三态门。整形电路是由一个运算放大器 μA741 构成的

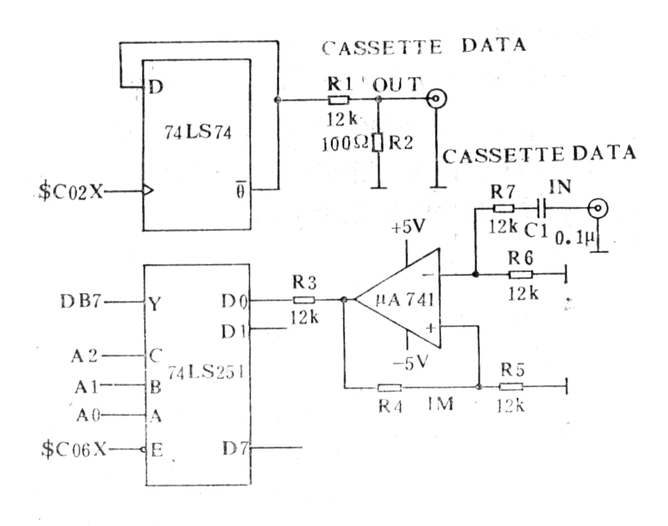


图 6-3 录音机输入、输出接口

带滞后反馈的过0比较器，输入信号约为 1V。比较器的输出为 ±4V 脉冲，通过电阻 R_3 和门输入电路，转换为 TTL 电平。用读 \$C060 的指令即可将这个 TTL 信号选通到数据总线 DB_7。计算机为了测量一个脉冲的宽度，它先找到一个脉冲边沿，然后计时并找另一个边沿，这样，一个脉冲宽度就测量到了。找到 200μs 宽度的同步头后，下面就是数据，就可读入文件了。

与盒带输出相似，喇叭发声口用一个 D 触发器接成分频器，不过时钟信号是来自 \$C03X，而且输出端有一个功率放大器。不同的音调、音色是用软件控制的。

图 6-4 中的8选1三态门的其它输入用于游戏棒的开关量输入和模拟量的输入。

§ 6.5 游戏棒接口

XMF-I 主机板上有一个游戏棒插座，允许从插座上输入三个开关量 SW_0、SW_1、SW_2，四个模拟量 PD_0、PD_1、PD_2、PD_3。此外，插座上还有一个选通信号输出供用户使用。游戏棒接口电路如图 6-4。

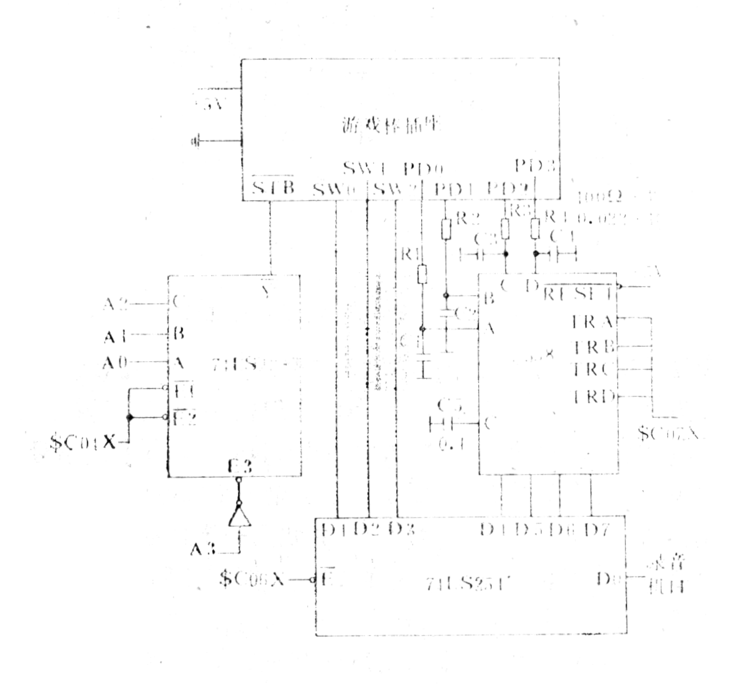


图 6-4 游戏棒接口

三个开关量 SW_0、SW_1、SW_2接在 74LS251 的 D_1 D_2、D_3 三个输入端（就是图 6-4 中的同一 74LS251）；所谓模拟量输入，是指四只电位器或压敏电阻之类。电位器

接在四个定时器 558 的四个定时电路中，与四个串联的小电阻和四个接地电容构成四个时间常数，决定四个定时器的延时时间。当定时器被来自 $C07X 的选通脉冲触发时，一个定时过程就开始了。通过软件测量定时时间变化，就可知道相关的电位器的阻值变化。这四个定时器的输出 QA、QB、QC、QD 接在 74LS251 其余四个输入端上，通过不同地址读到数据总线 DB7 上。

四个定时器的 $\overline{\text{RESET}}$ 接+5V，复位无效，控制端 C 不用，接一退耦电容使其工作稳定。

§ 6.6 打印机接口

主机板上设置了一个连接并行打印机的接口，打印机程序也已固化在 EPROM 中。打印接口是按照打印机数据传送方式设计的。这个接口向打印机传送 8 位锁存的数据和一个选通信号，同时有一位输入接口用于检测打印机的"BUSY"状态。这个接口占用 1 号扩展插槽。打印机接口如图 6-5。

当主机发一读 $C180（或 $C1C0）地址的命令时，打印机发出的"忙"状态（BUSY）通过三态门接上数据总线 DB7，如果读得 DB7=0，则表示打印机可以接收新的数据。此时，主机即可通过写 $C09X 的命令把数据送到 74LS273，并在 $C09X 选通脉冲的后沿开始锁存在它的 Q0～Q7，$C09X 选通脉冲后沿被延迟整形、微分再整形，作为 $\overline{\text{STROBE}}$ 脉冲送入打印机，告知它 Q0～Q7 上的锁存数据已稳定有效。主机与打印机的数据传送就这样进行下去。

汉字的打印机程序设置在汉字处理程序中。

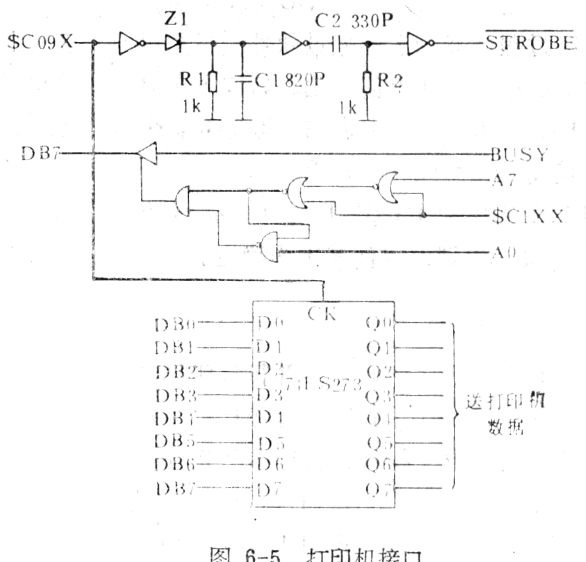


图 6-5 打印机接口

§6.7 扩展 I/O

XMF-I 可配接一个 I/O 扩展箱。箱内设有电源、一个软盘驱动器接口卡、两个软盘驱动器和四个用户可用的 I/O 扩展插槽。用户不能插入其它 I/O 接口的插槽是：

0 号槽：主机上已用于扩展 RAM 为 64K；

1 号槽：已用于主板上的打印机接口；

5 号槽：已用于主板上的汉字处理；

6 号槽：在扩展箱内用于软盘驱动接口卡。

四个为用户保留的扩展槽都是 50 芯插座。扩展一个用

户卡所需的总线信号、电源、时钟、地址、选通信号都已定义在引脚上。说明如下：

表 6-2

引　脚	信号名称	说　　明
1	$\overline{\text{I/O SEL}}$	输入信号。此线通常为高，当CPU访问 $CnXX 地址时（n为插槽号1～7），在Φ2周期变为低电平。
2～17	AD0～AD15	双向信号。16位地址总线。在Φ1期间变为有效，并在整个Φ2周期保持有效。
18	$\text{R}/\overline{\text{W}}$	双向信号。读/写。在Φ1期间变为有效，并在Φ2的整个期间保持有效。读为高电平，写为低电平。
19	SYNC	输入信号。视频同步。仅7号槽有。
20	$\overline{\text{I/O STB}}$	输入信号 I/O 选通。当访问地址为 $C800～$CFFF 时，在Φ2期间为低电平。
21	RDY	双向信号。就绪。在Φ1期间降为低电平。CPU仅在读周期内能识别RDY的低电平。
22	$\overline{\text{DMA}}$	输入信号。直接存储器访问。在Φ1开始时取低电平以停止处理并产生地址、数据和 $\text{R}/\overline{\text{W}}$ 的高阻态。
23	INT OUT	输出信号。中断输出。连到较低级插槽的菊形链。如果使用，通常输出高电平，如果不用，则连到引出线28。
24	DMA OUT	输出信号。直接存储器访问输出。连到优先级较低的菊形链。通常使用时，输出高电平；不使用时连到引出线27。
25	+5V	扩展箱内+5V电源。
26	GND	地线。
27	DMAIN	输入信号。直接存储器访问输入。从较高优先级来的菊形链输入，通常为高电平。

续表

引脚	信号名称	说　　明
28	INT IN	输入信号。中断输入。从优先级较高插槽来的菊形链输入通常为高电平。
29	$\overline{\text{NMI}}$	不可屏蔽中断输入信号。
30	$\overline{\text{IRQ}}$	中断请求。输出信号。取低电平启动 6052 的可屏蔽中断。仅当中断禁止标志复位时可以识别。
31	$\overline{\text{RESET}}$	复位。双向信号。作输出时仅低电平使 CPU 和外围设备复位。作输入时，在从键盘来的复位、加电时和从其它外围设备来的复位期间变为低电平。
32	$\overline{\text{INH}}$	输出信号。内部 ROM 禁止。取低电平禁止 ROM 地址空间 \$D000～\$DFFF。
33	－12V	扩展箱内电源－12V。
34	－5V	扩展箱内电源－5V。
35	COLO REF	输入信号。彩色参考信号。仅连至7号槽。
36	7M	输入信号。7.15909MHz 时钟。
37	Q3	输入信号。2.040968MHz(平均)时钟。
38	ϕ_1	输入信号 ϕ_1时钟。1.020484MHz（平均）系统时钟。
39	USER1	输出信号．若为低电平则禁止 I/O 地址译码 \$C000～\$C7FF。
40	Φ0	输入信号。1.020484MHz 0相时钟，与Φ1 相位相反。
41	$\overline{\text{DEV SEL}}$	输入信号。设备选择。当访问 \$C0（N+8）X 地址时，在Φ2 周期变低(N为插槽号)。每槽线选通 16 个地址。
42～49	D7～D0	双向信号。数据总线 D0～D7 数据在Φ2 期间变为有效直到Φ2 结束。
50	＋12V	扩展箱内电源＋12V。

上面的输入、输出或双向信号是对接口卡而言。

第七章　扩展箱及软磁盘驱动器接口

§7.1　概述

中华学习机“小蜜蜂”（XMF-I）型计算机采用积木式结构，可以根据不同用户的需要，灵活地构成各种配制、各种规摸的微机系统。积木式结构中的几个基本硬件除了主机、显示器、盒式磁带机外，还有扩展箱、打印机等设备可供选择。在这些外设中，扩展箱是比较重要的设备之一。

XMF-I 的扩展箱由下面三部分组成：

1. 为系统提供的软磁盘驱动器及主机与驱动器的接口电路。

2. “XMF-I”的外部总线与标准 APPLE-Ⅱ 扩展槽总线转换逻辑。

3. 供扩展箱及主机使用的稳压电源。

扩展箱与主机由一个称为 SW_2 的插头通过 50 芯扁平电缆连接。使用扩展箱时，要去掉原来的外接小电源，改由扩展箱系统电源经 50 芯电缆插头向主机直接供电。“XMF-I”扩展箱电源采用开关式稳压电源，提供四种电压，参数如下：

输入电压：交流 220V 或 110V

输出：	DC	+12V	2A
	DC	−12V	0.5A

DC	+5V	4A
DC	-5V	0.5A

扩展箱接口中的扩展槽接口信号是 APPLE-Ⅱ 的标准接口信号，使用时需要将主机 SW_2 的 50 条输入/输出信号转换成 APPLE-II 的标准扩展槽所需信号，其对应关系参见表 7-1。

主机通过驱动器接口电路实现对驱动器的控制，其框图如图 7-1 所示。

"XMF-I" 型计算机驱动器采用半高度超薄形驱动器，工作电压为 ±12V，±5V。驱动器接口电路由中、小规模集成电路实现，驱动器接口电路与扩展槽信号转换电路及标准扩展槽均做在同一块印刷电路板上，即扩展箱接口板。其组成如图 7-2 。

从图 7-2 可以看出，扩展箱本身作为一个外设使"XMF-I" 系统得到扩充，同时，扩展箱中的标准 APPLE-II 扩展槽又可插接 APPLE-II 上已开发的各种接口卡，从而进一步扩充系统功能。

"XMF-I" 型扩展箱中共提供了五个标准 APPLE-II 扩展槽，槽号分别为 2#，3#，4#，5#，7#。由于主机系统中，汉字占用了 5# 槽，驱动器接口电路占用 6# 槽，因此，实际可用的扩展槽只有四个：2#，3#，4#，7#。

下面对驱动器接口电路进一步作如下分析。

§ 7.2 器件介绍

扩展箱接口板的基本模块如图 7-2 所示，板上用到的几个主要器件有：

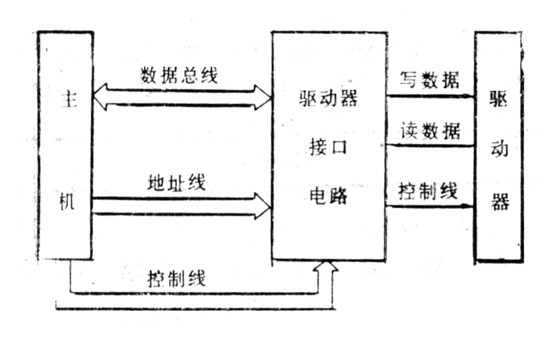


图 7-1 主机—驱动器连接框图

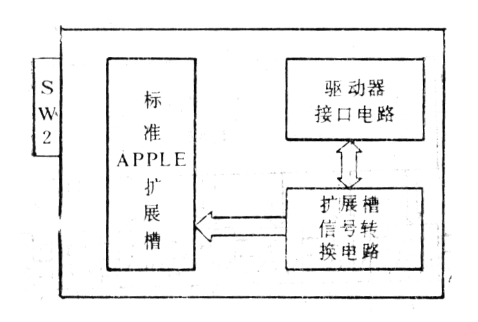


图 7-2 扩展箱接口板模块图

1． PROM

"XMF-I" 使用的 PROM 为三态输出低功耗 2K 位(256×8)可编程序只读存储器。其引脚排列如图 7-3 所示。

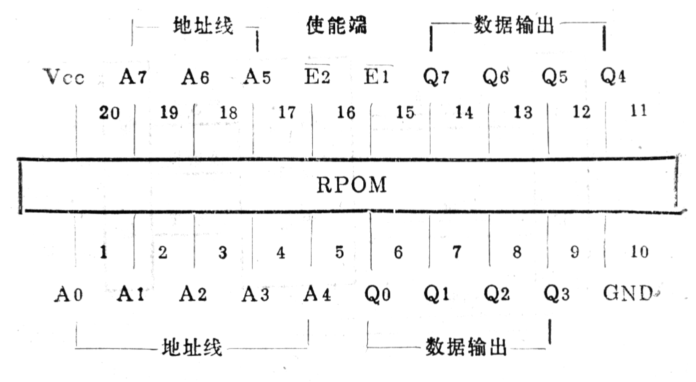


图 7-3 PROM 引脚排列图

该 PROM 的内部结构如图 7-4 所示。

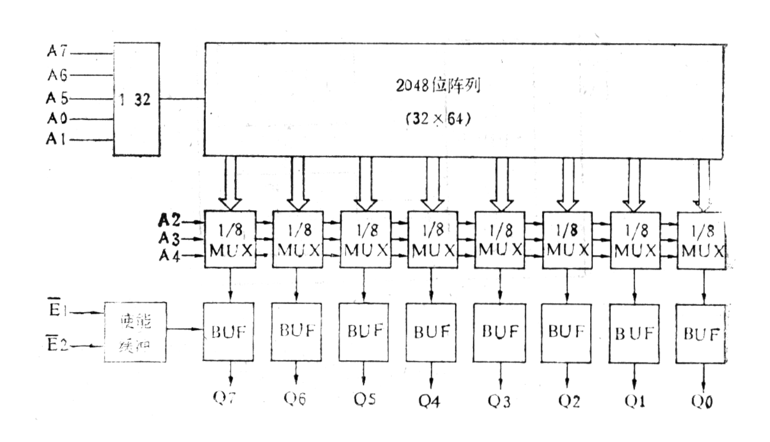


图 7-4 PROM 内部结构

从上图可以看到，PROM 的工作原理为：地址线 A0

～A7 选中 256 个字节中的某一个八位字节，当使能端 $\overline{E1}$、

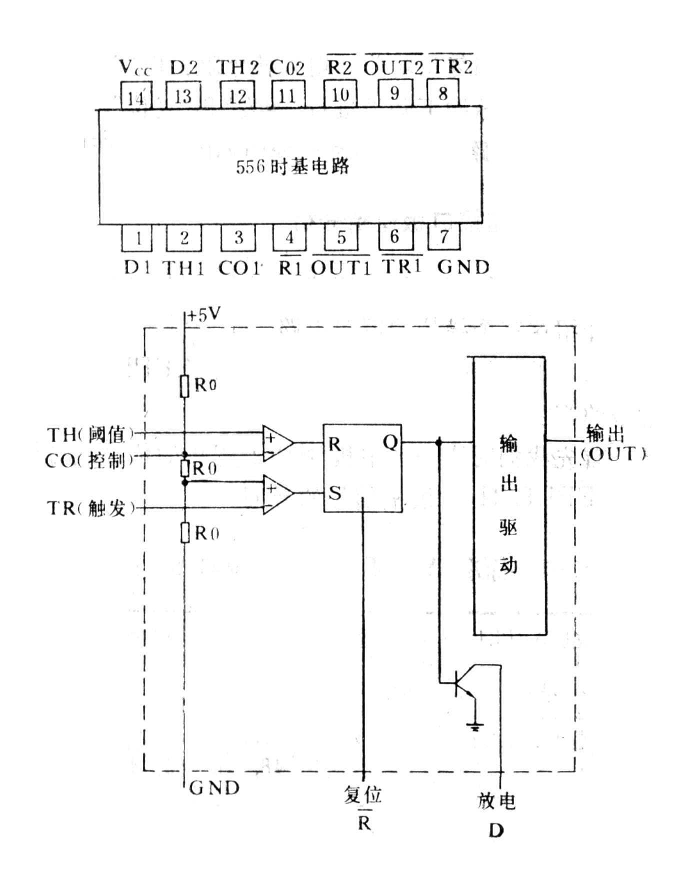


图 7-5 556 逻辑及引脚图

E_2 均为低电平时，这个字节的 8 位数据通过 Q0～Q7 输出。驱动器接口电路用一片 PROM 产生时序，以控制对磁盘的读写。

2. 时基电路 556

556 是一个能产生稳定而精确时间延迟和振荡频率的控制器，有触发和复位功能。用外接电阻、电容来控制时间延迟。556 利用脉冲下降沿进行“触发”和“复位”，输出可直接驱动 TTL 电路，其引脚及原理图如图 7－5 所示。

§ 7.3 扩展箱接口板逻辑分析

前面已经提到，扩展箱接口板由三部分组成，下面我们分别介绍这三个模块的逻辑电路。

1. 扩展槽信号转换电路及标准 APPLE-II 扩展槽信号的产生。

首先我们对比一下主机 SW2 提供的信号线与标准 50 芯 APPLE-II 扩展槽信号的异同。

表 7-1 标准 APPLE-Ⅱ 扩展槽与 SW2 信号对照表

标准 APPLE-II 扩展槽信号		对应的主机 SW2 信号
信号名	引脚号	引脚号
I/O SELECT	1	I/O槽对应的信号由47，3，2，48，46，45，4给出
16位地址线	2～17	16 条地址线分别由 24～17，12，11，16，15，6，7，8，5 给出
读/写线 $R/\overline{W}$	18	43

续表

I/O STROBE	20	37
SYNC(7#槽)	19	无
RDY	21	41
$\overline{DMA}$	22	36
INT OUT	23	无
DMA OUT	24	无
+5V	25	9,50
GND	26	1,26
DMA IN	27	无
INT IN	28	无
$\overline{NMI}$	29	42
$\overline{IOQ}$	30	49
$\overline{RES}$	31	14
$\overline{INH}$	32	13
−12V	33	无
−5V	34	25
COLOR REF (7#槽)	35	无
7M	36	39
Q3	37	40
Φ1	38	44
USER1	39	35
Φ0	40	38
DEVICE SELECT	41	自译
D7～D0	42～49	27～34
+12V	50	无

从表中可以看出，“XMF-I”主机的 SW_2 信号与标准 APPLE-II 扩展槽总线信号转换时，只需再提供 I/O

SELECT、DEVICE SELECT 这两组信号，其它信号都可一一对应，直接相连。DMA IN，DMA OUT 和 INT IN，INT OUT只规定外设（即各接口卡）之间的 DMA 与中断顺序，而与主机系统无关。至于 ±12V 和 ±15V，由于扩展箱本身就有电源，因此，可直接由扩展箱电源提供。

I/O SELECT 信号由 SW_2 给出。在主机中，已将这几条信号线译出，分别起名为 I/O_0，I/O_1……I/O_7，因而，只需把对应的 I/O SELECT 与对应的扩展槽相连。

$\overline{\text{DEVICE SELECT}}$ 由扩展箱接口板上的一片 3-8 译码器 74LS138 完成，其逻辑图如图 7－6 。

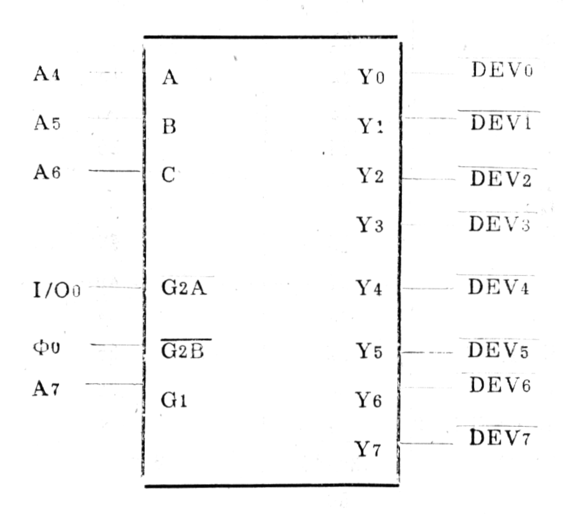


图 7-6 DEVICE SELECT 信号产生电路

标准 APPLE-II 扩展槽中 DEVICE SELECT 产生的

条件为：当地址总线的地址为 $C0n0 至 $C0nF 时，由编号 n 所决定的某个扩展槽的地址选通信号为低电平，同时要求该信号在 Φ_0 的正半周有效，n 值取决于 I/O_0与 A_4~A_7 地址的组合。当 I/O_0 有效时，其口地址为 C0XX，它与 A_4~A_7 共同组合，分别译出 C08X~C0FX，对应的译码输出结果即为 $\overline{\text{DEVICE SELECT}}$，我们分别定义为 $\overline{DEV_0}$~$\overline{DEV_7}$。

在信号转换电路中，为了改善信号的品质，SW_2 与 APPLE-Ⅱ 扩展槽连接的 8 位数据线和 16 位地址线分别加上 4.7K 的提拉电阻，其余信号直接相连。

2． 驱动器接口电路

“XMF-I”型软磁盘驱动器电路可驱动两个磁盘机，使用 APPLE-Ⅱ DOS 3.3 操作系统时，每个格式化后的软盘共有 35 个磁道，每个磁道 16 个扇区，每个扇区 256 个字节，总容量为 140KB。

磁盘机工作过程如下：

磁盘机占用 6 号槽口，在 XMF-BASIC 下执行 PR#6，或监控下执行 C600G 均可启动磁盘驱动器。启动磁盘驱动器后，系统读入引导程序，接着转去执行引导程序，再由引导程序调入操作系统。此后，可通过操作系统命令对软磁盘上的文件进行读写或执行。我们这里要介绍的是如何启动磁盘驱动器，而不去讨论程序如何调出操作系统。上述过程如图 7－7 所示。

启动磁盘机读入引导程序的过程包括如下三个动作：

（1） 磁头复位，即磁头回到第 0 道。

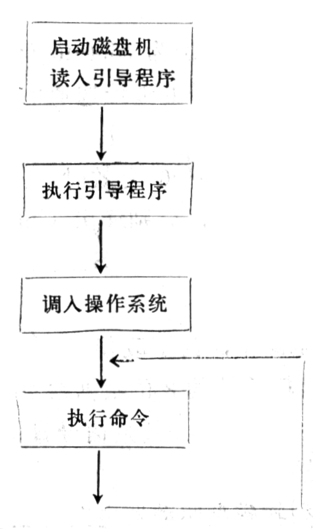


图 7-7 “XMF-I”磁盘工作过程

(2) 从 0 道 0 扇区读入引道程序，并放到 $801 起始的内存单元。

(3) 引导程序读完后，转去从 $801 开始执行程序。

为了从 0 道 0 扇区开始执行程序，在 $C600～$C6FF

表 7-2 软开关功能及地址表

地址	功能
$C0E0	关闭步进马达第 0 相位
$C0E1	打开步进马达第 0 相位
$C0E2	关闭步进马达第 1 相位
$C0E3	打开步进马达第 1 相位
$C0E4	关闭步进马达第 2 相位
$C0E5	打开步进马达第 2 相位

续表

$C0E6	关闭步进马达第3相位
$C0E7	打开步进马达第3相位
$C0E8	关闭马达
$C0E9	打开马达
$C0EA	起动第一号磁盘机
$C0EB	起动第二号磁盘机
$C0EC	送出脉冲，表示要作输入/输出
$C0ED	将数据送入磁盘机中的锁存器
$C0EE	令锁存器进入输入准备状态
$C0EF	令锁存器进入输出准备状态

$C0EE，$C0ED 同时使用，相当于感应磁盘是否有防止写入的功能。

$C0EE，$C0EC 同时使用，相当于读取。

$C0EF，$C0EC 同时使用，即写入。

这 256 个字节的内存地址空间中固化了启动程序，这部分程序的功能就是完成上述三个操作。这部分程序可以固化在 ROM 上，放在驱动器接口电路中，也可以写在 EPROM 中。“XMF-I”型将这部分程序写在主机上的 64K 位 EPROM2764 中。启动程序通过 16 个软开关控制驱动器接口电路，这 16 个软开关分成 8 组，每组各有 2 个，分别控制开和关。软开关的地址与当前的接口电路所在的槽号有关，若假定插在6# 扩展槽（此时，软开关地址为 $C080＋60～$C08F＋60），则各开关的地址及功能如表 7－2。

合理地使用这些开关，可对驱动器接口电路进行控制。换句话说，驱动器接口电路必须在这些开关控制下才能正常工作。

接口电路的逻辑框图如图 7－8。

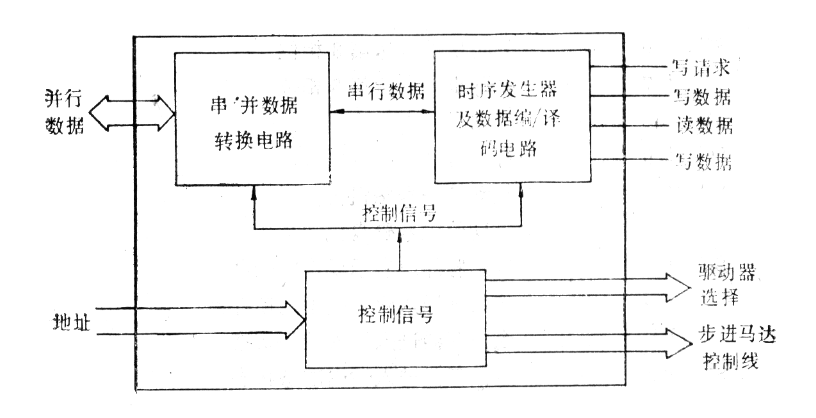


图 7-8 驱动器接口逻辑框图

驱动器接口电路与驱动器通过一条 20 芯扁平电缆连接，每个驱动器用一条，其定义如表 7－3：

表 7-3 驱动器与驱动器接口电路 20 芯扁平电缆定义

信号名	信号功能	引脚号
+5V		11，12
−12V		9
−12V		13，15，17，19
GND		1，3，5，7
RD DATA	读入数据	16
WR DATA	写数据	18
WR REQ	写请求	10
WR PROT	有写保护标记时为 1	20
ENBL1	选驱动器 1	14a

续表

ENBL2	选驱动器2	14b
Φ0	相位0控制线	8
Φ1	相位1控制线	6
Φ2	相位2控制线	4
Φ3	相位3控制线	2

串并数据转换由一片 74LS323 完成。由于磁盘机要求读/写数据为串行数据，因此，读出或写入都必须经过串并转换。串并转换的转换频率为主机提供的 2MHz 非对称时钟 Q3。

时序发生器及数据编码/解码器由一片 256×8 的 PROM 和一片 6D 触发器 74LS174 完成。74LS174 与 PROM 构成一反馈回路，产生读写所需的时序，该时序的变化情况还受到某些软开关的控制。

控制信号的产生主要是通过一片 74LS259，由它产生控制马达相位的控制信号和控制时序电路的控制信号。"XMF-I"型微型机的寻道过程靠控制步进马达的相位实现。要移动磁盘驱动器中的磁头，必须依次打开，然后关闭步进马达的四个相位。若按由小到大（0～3）的顺序开关，磁头向圆心移动，若按由大到小的顺序开关，磁头向圆周移动。

由于磁盘机的时序要求比较苛刻，在程序设计和硬件电路中要求必须在 32 个机器周期中写完，在设置了写入方式后，需延迟 100 微秒才能开始写入。因此，在硬件中，选用 556 进行时序控制，以求得较高的控制精度。

第八章　汉字系统逻辑电路

§ 8.1　概述

为了使微型计算机系统能够更好地适合我国中、小学及家庭的需要，中华学习机“小蜜蜂-I”（XMF-I）微型计算机配备了功能较强的汉字系统。采用叠加法压缩字库及其恢复技术，固化了全部的国标一级和二级简化汉字，共有6763个。同时，还可以处理、显示和打印与其对应的繁体字。XMF-I微型计算机的汉字系统配有六种输入方法，即：国标、区位、声韵、简易拼音、偏旁部首及全拼音输入方法，可以适合各种操作者的不同需要。此外，还配有2500～7000个联想式词组，从而提高了汉字的输入速度。

“XMF-I”微型计算机的汉字系统是由Z80A CPU、Z80 PIO、EPROM(27256，2764)、静态RAM 6116以及译码控制电路组成，其系统组成框图如图8-1所示。

其中，Z80A CPU为汉字系统的处理机，Z80 PIO是汉字系统与65SC02系统的接口，Z80A CPU将其初始化成双向方式，它负责接收由65SC02系统送来的编码，并以中断方式送给Z80A CPU去处理，然后，将Z80A CPU送出的汉字点阵传送到65SC02系统。

在“XMF-I”型微型计算机汉字系统中有三片EPROM 27256和一片2764，以及一片静态RAM 6116。其

中两片 27256 中存有汉字字库及其恢复程序，可以有5种输入方法，即：国标、区位、简易拼音、全拼音及偏旁部首，并包含有 7000 个词组。另一片 27256 是一个可选件，如果选用，除上述功能外，还可增加声韵编码输入法。RAM 6116 用作 Z80A CPU 的缓冲工作单元，用来存放随机选中的汉字恢复点阵。以上4片存储器所占的物理地址为 96K，但是 Z80A CPU 的直接寻址能力只有 64K，所以，采用了低 32K 两体切换的办法，由 Z80A CPU 来控制寻址。

在 EPROM 2764 中固化了用 65SC02 汇编语言写的汉字系统处理程序，还有 65SC02 系统中的打印机和软磁盘驱动器的处理程序和引导程序。这片 EPROM 2764 由

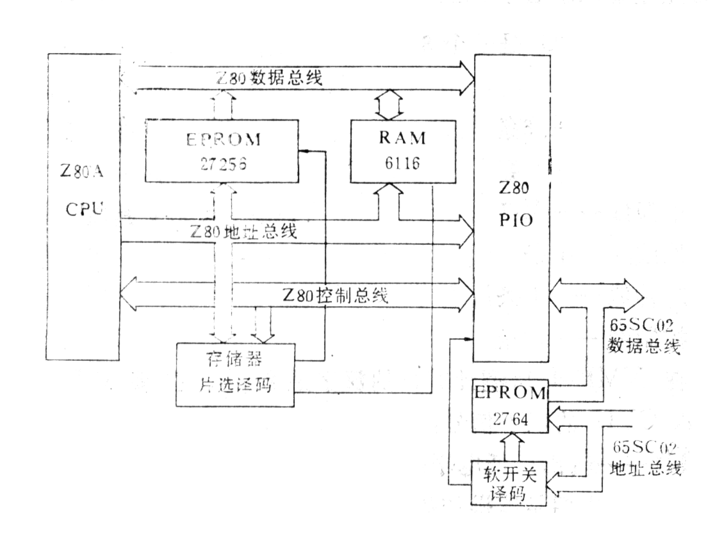


图 8-1　汉字系统组成框图

65SC02 CPU 来寻址和控制。

汉字系统占用 XMF-I 计算机扩展槽的第 5 号槽口控制信号。利用扩展 I/O 地址 \$C800～\$CFFF 进行引导。因此，可以在 BASIC 状态下直接键入 PR#5 来进入汉字系统，

§ 8.2 汉字系统电路

主要器件及其信号

1) Z80A CPU

这是汉字系统的处理机。汉字系统的一切操作都在Z80A CPU 的控制之下进行，如接收编码、查找字库和码本、运行汉字字形的恢复程序等。

Z80A CPU 是一个 8 位微处理器，有 40 条引腿，其中

地址线——16条

数据线—— 8 条

电源线—— 1 条

地线 —— 1 条

时钟线—— 1 条

控制线——13条

在 XMF-I 微型计算机的汉字系统中，主要使用了 Z80A CPU 的以下信号：

$\overline{M1}$——输出，低电平有效。在 $\overline{M1}$ 为低电平时，表示现行的机器周期是取操作码周期。

$\overline{MREQ}$ ——三态输出，低电平有效。这是一个存储器

请求信号。$\overline{\mathrm{MREQ}}$ 为低电平表示地址总线上保持有一个供存储器读或写的有效地址。

$\overline{\mathrm{IORQ}}$ ——三态输出，低电平有效。这是一个输入/输出请求信号。当 $\overline{\mathrm{IORQ}}$ 为低电平时，表示地址总线的低8位中保持有一个供 I/O 读或写的端口地址。同时，$\overline{\mathrm{IORQ}}$ 还有另一个作用，就是当中断被响应时，$\overline{\mathrm{IORQ}}$ 和 $\overline{\mathrm{M1}}$ 一起变为有效状态，表示响应中断，通知外设把中断矢量放到数据总线上。

$\overline{\mathrm{RD}}$ ——读信号，表示 CPU 要求从所寻址的存储器或 I/O 设备读入数据。

$\overline{\mathrm{WR}}$——写信号，表示 CPU 数据总线上保持了有效的数据，要求写入所寻址的存储器或 I/O设备。

$\overline{\mathrm{INT}}$——中断请求信号，输入，低电平有效。这个信号由 I/O 设备来产生。CPU 响应中断的条件有两个：一个是在 Z80A CPU 内部有一个由软件控制的中断允许触发器，它必须是处于开中断状态；另一个条件是没有总线请求。如具备这两个条件，那么，Z80A CPU 在现行指令周期结束之后响应中断，在下一个指令周期的开始送出一个中断响应信号，即 $\overline{\mathrm{M1}}$ 与 $\overline{\mathrm{IORQ}}$ 同时为低电平。

（2）Z80 PIO

Z80 PIO 是一个可编程序的并行输入、输出接口部件，其主要的功能有：

1）具有两个独立的8位双向输入、输出端口，而且具有相应的控制和状态信号。

2）每个端口都有以下几种工作方式：

方式 0 ——输出方式。

方式 1 ——输入方式。

方式 2 ——双向方式（只有端口 A 可用）。

方式 3 ——位控方式。

3）能自动提供中断矢量，不需要外加逻辑。

另外，Z80 PIO 还有一个突出特点，即外设与 CPU 之间的全部数据传送都是在中断控制之下完成的。

在 XMF-I 计算机汉字系统中，主要使用了 Z80 PIO 的以下控制信号：

B/$\overline{A}$ ——端口选择。低电平选端口 A，高电平选端口 B。

C/$\overline{D}$——控制字或数据选择。本引腿规定了在 CPU 和 PIO 之间进行信息传送的类型。当C/$\overline{D}$ 为高电平时，CPU 写入 PIO 的是一个控制命令，为低电平时，CPU 与 PIO 之间是数据传送。

$\overline{CE}$ ——片选允许信号，低电平有效。

$\overline{M1}$，$\overline{IORQ}$，$\overline{RD}$——这三个信号都来自 Z80A CPU，它们必须配合使用。在$\overline{CE}$有效时，当 $\overline{M1}$ 有效而 $\overline{IORQ}$ 和 $\overline{RD}$ 信号无效（均为高电平）时，PIO 进入复位状态。当$\overline{IORQ}$ 和 $\overline{M1}$同时为低电平时，表明 CPU 对 PIO 的中断请求给予响应。当 $\overline{IORQ}$ 和 $\overline{RD}$ 信号都是低电平时，表示 CPU 从 PIO 读取数据。当 $\overline{IORQ}$ 为低、$\overline{RD}$ 为高时，表明 CPU 向 PIO 写控制字或数据。

$\overline{INT}$——中断请求信号。当 $\overline{INT}$ 为低电平时，表示 PIO 向 Z80A CPU 申请中断。

在 XMF-I 微型计算机汉字系统中，Z80 PIO 工作在双向方式下，端口 A 既可以输入，又可以输出。输出控制使用端口 A 的连络信号 $\overline{ASTB}$ 和 $\overline{ARDY}$，输入控制用端口 B 的连络信号 $\overline{BSTB}$ 和 $\overline{BRDY}$。

2. 汉字系统存储器译码电路

汉字系统的存储器地址分配如图 8－2 所示。

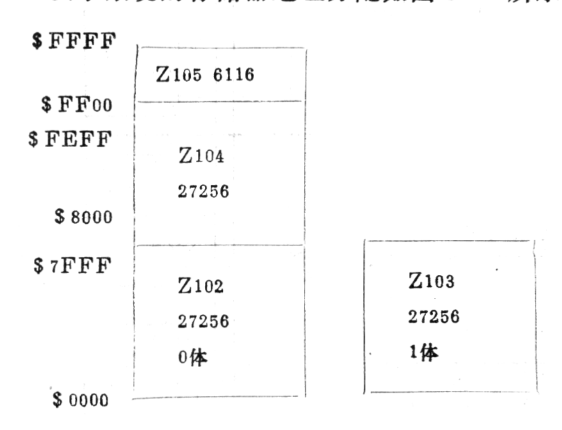


图 8-2 汉字系统存储器地址分配图

从图中可以看出，两片 27256 的地址是相同的，都是处在低 32K 范围内。这是因为汉字系统所用的存储器为 96K，但 Z80A CPU 的寻址范围只有 64K，所以采用了低 32K 两体切换的方法来解决。

实现上述地址分配要求的逻辑电路如图 8－3。

在低 32K 范围内，分 0 体和 1 体，这两个体的切换由 Z80 PIO 的 PB0 线来控制。

系统中规定：

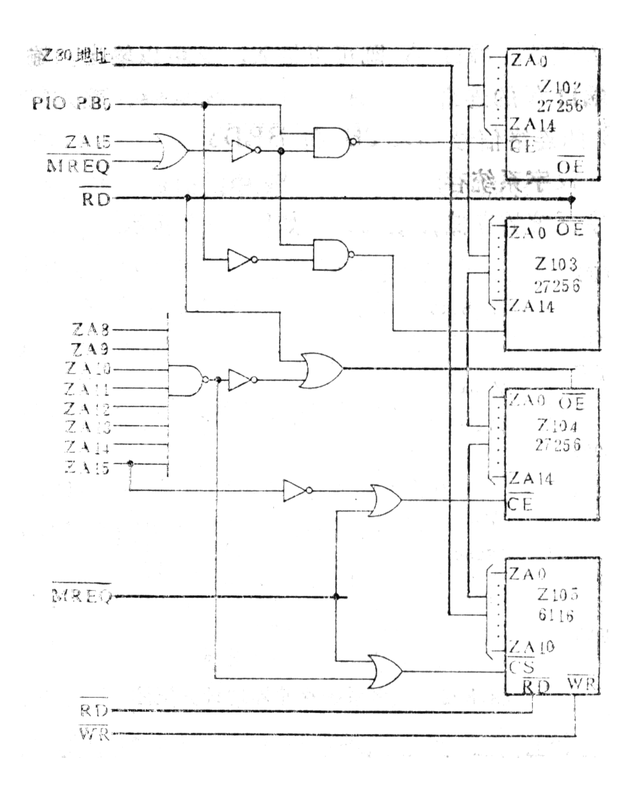


图 8-3 汉字系统转体译码逻辑电路

PB0 为高电平选 0 体，即选 Z102。

PB0 为低电平选 1 体，即选 Z103。

它们的地址范围都是 \$0000～\$7FFF，在这个范围内，Z80A 地址线 A15 总是低电平。

高 32K 字节的地址分配给 Z104 和 Z105，其中 Z105

所占的地址范围是 $FF00～$FFFF，Z104 所占的地址为 $8000～$FEFF。这个分配通过高 8 位地址的译码来实现。

3. EPROM 2764 地址译码

在 XMF-I 微型计算机的汉字系统中，用了一片 EPROM 2764 来固化 4K 的汉字处理程序，占用高 4K 地

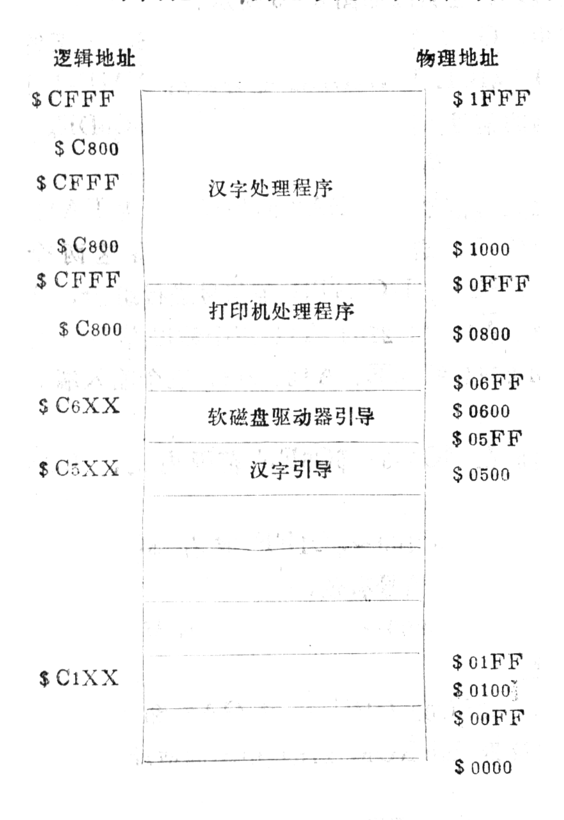


图 8-4 EPROM 2764 地址分配图

址。另外，还有打印机和软磁盘驱动器的引导程序，分别占用 $C100～$C1FF 和 $C600～$C6FF 两段地址。EPROM 2764 的详细地址分配情况见图 8－4。

实现这一地址分配的地址译码工作由门阵列器件 GA7 和 GA8 通过控制 2764 的片选及 A11、A12 两位高位地址来实现。其逻辑图如图 8－5 所示。

其中，GA7 的输入 A0～A3 是65SC02 系统的地址信号。DEVICE ENABLE 是分配给 XMF-I 计算机 5 号扩展槽的信号，其地址码为 $C0D0～$C0DF。R/W 为 65SC02 的读写信号。

以上几种信号经过 GA7 的处理，产生 PA11 和 PCE 两个输出信号，送至 GA8。GA8 再将这两个信号与 C5XX、BPEN、I/O STB 等信号进行组合，产生 2764 CE，2764 A12 和 2764 A11 三个输出信号，直接送到 EPROM 2764 的 CE、A12、A11 三个输入端，控制地址变换。

当地址在 $C500～$C5FF 范围内时，C5XX 信号有效；

当地址在 $C100～$C1FF 或者 $C600～$C6FF 范围内时，BPEN 信号有效；

当地址在 $C800～$CFFF 这 2K 范围内时，I/O STB 信号有效。

这样，65SC02 CPU 通过 GA7 和 GA8 的控制和变换，来读取 EPROM 2764 中不同地址范围内的程序和数据。

例如，当键入 PR#5 时，计算机要执行 $C500～$C5

FF 内的程序。这时，$\overline{\text{C5XX}}$ 有效，经过 GA8 的变换，使 2764 CE、2764 A12 和 2764 A11 三个输出端都是低电平，这样，地址便落在 EPROM 2764 的最低 2K 范围内，再根据地址总线上 A10～A0 的状态，选中地址为 \$C500～\$C5FF 的256 个单元，运行汉字引导程序，使计算机进入汉字处理状态工作。

4. 汉字系统与 65SC02 系统间的信息传送

前已述及，Z80A CPU 为汉字系统的处理机，而 Z80 PIO 是两个处理机之间的接口部件。在汉字处理状态下，65SC02 CPU 要通过 Z80 PIO 将编码送入 Z80A CPU，并通过 PIO 得到点阵及其它信息。有关的电路见图 8-6。

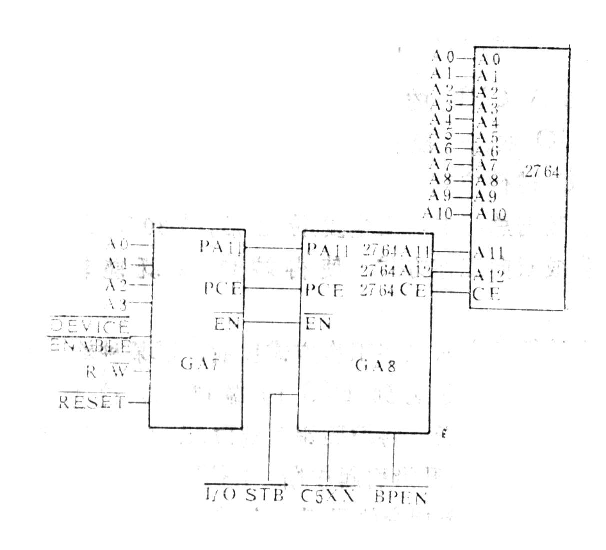


图 8-5 EPROM 2764 译码逻辑电路

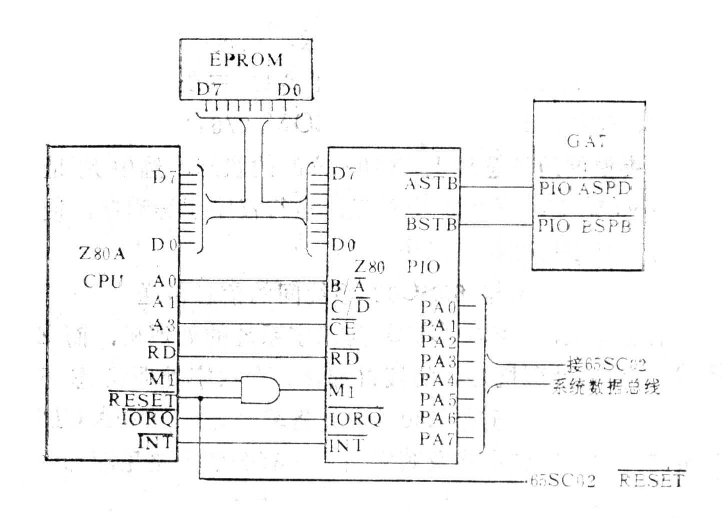


图 8-6 汉字系统与 65SC02 系统通讯电路

Z80A CPU 的 3 根地址线 A0、A1、A3 分别接到 Z80 PIO 的三个控制引腿 $B/\overline{A}$、$C/\overline{D}$ 和 $\overline{CE}$ 端，以便对 PIO 进行控制。Z80 PIO 的端口 A 直接连到 65SC02 系统的数据总线上。上电复位之后，Z80A CPU 将 PIO 初始化成双向工作方式，使得端口 A 既能输入又能输出。

当计算机进入汉字工作方式以后，以某种输入方法从键盘上敲入的编码送到 PIO 的端口 A，这时，GA7 的 $\overline{\text{PIO BSTB}}$ 变低，将编码锁入端口 A 的输入寄存器。$\overline{\text{PIO BSTB}}$的上升沿引起中断，$\overline{\text{INT}}$ 变为低电平。Z80A CPU 在执行一条指令的最后一个 T 周期采样 $\overline{\text{INT}}$，并送出一个中断应答信号，$\overline{\text{M1}}$ 和 $\overline{\text{IORQ}}$ 都变为低电平。然后，

PIO 端口 A 将中断矢量放到 Z80 数据总线上，Z80A CPU 执行中断服务程序，将端口 A 输入寄存器中的数据读到 CPU 中。

在此之后，Z80A CPU 进行查找字库的工作，并运行恢复程序及其它有关程序。运行完毕，将汉字的点阵及有关信息送到 PIO，这时，$\overline{\text{RD}}$ 为高电平，$\overline{\text{IORQ}}$ 为低电平，CPU 将数据写到 PIO 的输出数据寄存器。65SC02 CPU 经一段等待时间之后，通过 GA_7 发出 $\overline{\text{PIO ASTB}}$ 信号，从端口 A 将数据读走，同时产生中断信号 $\overline{\text{INT}}$，完成了一次 PIO 数据输出的过程。Z80A CPU 在中断服务程序中再继续输出新的数据。这样，连续输出 32 个字节的数据构成一个汉字的点阵。

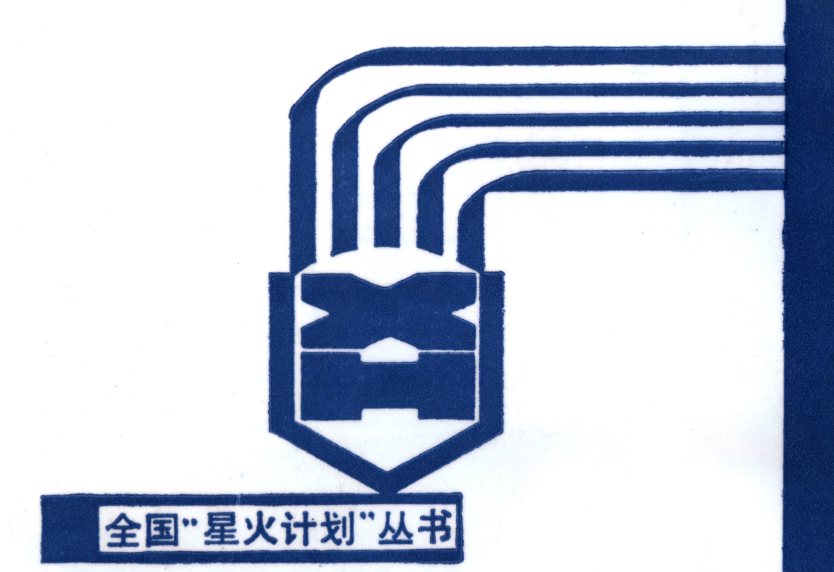
65SC02 CPU 指令表

序号	助记符	操作	隐含			累加器			立即数			零页			零页X变址			零页Y变址			绝对		
			OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN
1	ADC+	A+M+C → A 2							69	2	2	65	2	3	75	2	4				6D	3	4
2	AND+	A∧M → A							29	2	2	25	2	3	35	2	4				2D	3	4
3	ASL	C ← 7.6.5.4.3.2.1.0 ← 0				0A	1	2				06	2	5	16	2	6				0E	3	6
4	BCC	若C=0，则转移 1																					
5	BCS	若C=1，则转移 1																					
6	BEQ	若Z=0，则转移 1																					
7	BIT+	A∧M→Z, M_7→N, M_6→V							89+	2	2	24	2	3	34+	2	4				2C	3	4
8	BMI	若N=1，则转移 1																					
9	BNE	若Z=0，则转移 1																					
10	BPL	若N=0，则转移 1																					
11	BRA++	无条件转移																					
12	BRK	PC+2 状态寄存器P推入栈 FFFE→PCL FFFF→PCH	00	1	7																		
13	BVC	若V=0，则转移 1																					
14	BVS	若V=1，则转移 1																					
15	CLC	0 → C	18	1	2																		
16	CLD	0 → D	D8	1	2																		
17	CLI	0 → I	58	1	2																		
18	CLV	0 → V	B8	1	2																		
19	CMP+	A−M → N, Z, C							C9	2	2	C5	2	3	D5	2	4				CD	3	4
20	CPX	X−M → N, Z, C							E0	2	2	E4	2	3							EC	3	4
21	CPY	Y−M → N, Z, C							C0	2	2	C4	2	3							CC	3	4
22	DEC+	A−1→A 或 M−1→M				3A+	1	2				C6	2	5	D6	2	6				CE	3	6
23	DEX	X−1 → X	CA	1	2																		
24	DEY	Y−1 → Y	88	1	2																		
25	EOR+	A∀M → A							49	2	2	45	2	3	55	2	4				4D	3	4
26	INC+	A+1→A 或 M+1→M				1A+	1	2				E6	2	5	F6	2	6				EE	3	6
27	INX	X+1 → X	E8	1	2																		
28	INY	Y+1 → Y	C8	1	2																		
29	JMP+	(PC+1)→PCL (PC+2)→PCH																			4C	3	3
30	JSR	PC+2推入栈 (PC+1)→PCL (PC+2)→PCH																			20	3	6
31	LDA+	M → A							A9	2	2	A5	2	3	B5	2	4				AD	3	4
32	LDX	M → X							A2	2	2	A6	2	3				B6	2	4	AE	3	4
33	LDY	M → Y							A0	2	2	A4	2	3	B4	2	4				AC	3	4
34	LSR	0→76543200→C				4A	1	2				46	2	5	56	2	6				4E	3	6
35	NOP	不操作	EA	1	2																		
36	ORA+	A∨M → A							09	2	2	05	2	3	15	2	4				0D	3	4
37	PHA	推A入栈 S−1 → S	48	1	3																		
38	PHP	推状态寄存器P入栈 S−1→S	08	1	3																		
39	PHX++	推X入栈 S−1→S	DA+	1	3																		
40	PHY++	推Y入栈 S−1→S	5A+	1	3																		
41	PLA	栈顶内容弹出到A, S+1→S	68	1	4																		
42	PLP	栈顶内容弹出到状态寄存器 S+1→S	28	1	4																		
43	PLX++	栈顶内容弹出到X, S+1→S	FA+	1	4																		
44	PLY++	栈顶内容弹出到Y, S+1→S	7A+	1	4																		
45	ROL	[7←0]←[C]←				2A	1	2				26	2	5	36	2	6				2E	3	6
46	ROR	→[C]→[7→0]				6A	1	2				66	2	5	76	2	6				6E	3	6
47	RTI	栈顶内容依次弹出到P, PC, S+1→S	40	1	6																		
48	RTS	栈顶内容弹出到PC, PCH→PC SH→S	60	1	6																		
49	SBC+	A−M−C̄ → A 2							E9	2	2	E5	2	3	F5	2	3				ED	3	4
50	SEC	1 → C	38	1	2																		
51	SED	1 → D	F8	1	2																		
52	SEI	1 → I	78	1	2																		
53	STA+	A → M										85	2	3	95	2	4				8D	3	4
54	STX	X → M										86	2	3				96	2	4	8E	3	4
55	STY	Y → M										84	2	3	94	2	4				8C	3	4
56	STZ++	00 → M										64+	2	3	74+	2	4				9C+	3	4
57	TAX	A → X	AA	1	2																		
58	TAY	A → Y	A8	1	2																		
59	TRB++	A∧M → M										14+	2	5							1C+	3	6
60	TSB++	A∨M → M										04+	2	5							0C+	3	6
61	TSX	S → X	BA	1	2																		
62	TXA	X → A	8A	1	2																		
63	TXS	X → S	9A	1	2																		
64	TYA	Y → A	98	1	2																		
			隐含			累加器			立即数			零页			零页X变址			零页Y变址			绝对		

序号	助记符	绝对X变址			绝对Y变址			相对			绝对间址			零页X变址间址			零页间址Y变址			零页间址			绝对X变址间址			状态标志 7	6	5	4	3	2	1	0	指令类别
		OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	OP	BN	CN	N	V		B	D	I	Z	C	
1	ADC+	7D	3	4*	79	3	4*							61	2	6	71	2	5*	72*	2	5				✓	✓					✓	✓	带进位加法
2	AND+	3D	3	4*	39	3	4*							21	2	6	31	2	5*	32*	2	5				✓						✓		逻辑与
3	ASL	1E	3	6																						✓						✓	✓	算术左移
4	BCC							90	2	2																								条件转移
5	BCS							B0	2	2																								条件转移
6	BEQ							F0	2	2																								条件转移
7	BIT+	3C+	3	4*																						M_7	M_6					✓		程序控制
8	BMI							30	2	2																								条件转移
9	BNE							D0	2	2																								条件转移
10	BPL							10	2	2																								条件转移
11	BRA++							80+	2	3																								无条件转移
12	BRK																												1	0	1			软件中断
13	BVC							50	2	2																								条件转移
14	BVS							70	2	2																								条件转移
15	CLC																																0	程序控制
16	CLD																													0				运算指令
17	CLI																														0			程序控制
18	CLV																										0							程序控制
19	CMP+	DD	3	4*	D9	3	4*							C1	2	6	D1	2	5*	D2	2	5				✓						✓	✓	运算中比较指令
20	CPX																									✓						✓	✓	运算中比较指令
21	CPY																									✓						✓	✓	运算中比较指令
22	DEC+	DE	3	7																						✓						✓		运算器减1指令
23	DEX																									✓						✓		运算器减1指令
24	DEY																									✓						✓		运算器减1指令
25	EOR+	5D	3	4*	59	3	4*							41	2	6	51	2	5*	52+	2	5				✓						✓		逻辑"异或"
26	INC+	FE	3	7																						✓						✓		运算中加1指令
27	INX																									✓						✓		运算中加1指令
28	INY																									✓						✓		运算中加1指令
29	JMP+										6C	3	6										7C+	3	6									无条件转移
30	JSR																																	转子指令
31	LDA+	BD	3	4*	B9	3	4*							A1	2	6	B1	2	5*	B2+	2	5				✓						✓		数据传送
32	LDX				BE	3	4*																			✓						✓		数据传送
33	LDY	BC	3	4*																						✓						✓		数据传送
34	LSR	5E	3	6																						0						✓	✓	逻辑右移
35	NOP																																	空操作
36	ORA+	1D	3	4*	19	3	4*							01	2	6	11	2	5*	12+	2	5				✓						✓		逻辑"或"
37	PHA																																	数据传送
38	PHP																																	数据传送
39	PHX++																																	数据传送
40	PHY++																																	数据传送
41	PLA																									✓						✓		数据传送
42	PLP																									✓	✓		1	✓	1	✓	✓	数据传送
43	PLX++																									✓						✓		数据传送
44	PLY++																									✓						✓		数据传送
45	ROL	3E	3	6*																						✓						✓	✓	循环左移
46	ROR	7E	3	6*																						✓						✓	✓	循环右移
47	RTI																									✓	✓		1	✓	1	✓	✓	中断返回
48	RTS																																	返回指令
49	SBC+	FD	3	4*	F9	3	4*							E1	2	6	F1	2	5*	F2+	2	5				✓	✓					✓	✓	运算中减法指令
50	SEC																																1	程序控制
51	SED																													1				程序控制
52	SEI																														1			程序控制
53	STA+	9D	3	5	99	3	5							81	2	6	91	2	6	92+	2	5												数据传送
54	STX																																	数据传送
55	STY																																	数据传送
56	STZ++	9E+	3	5																														数据传送
57	TAX																									✓						✓		数据传送
58	TAY																									✓						✓		数据传送
59	TRB++																															✓		逻辑"或"
60	TSB++																															✓		数据传送
61	TSX																									✓						✓		数据传送
62	TXA																									✓						✓		数据传送
63	TXS																																	数据传送
64	TYA																									✓						✓		数据传送
		绝对X变址			绝对Y变址			相对			绝对间址			零页X变址间址			零页间址Y变址			零页间址			绝对X变址间址			N	V		B	D	I	Z	C	

说明：* 增址进位时CN+1　++ 65S02增加指令　+ 65S02附加寻址方式　CN 周期数　BN 字节数

封面设计：于　晏



ISBN 7-302-00274-6

TP·104　　定价：3.80 元

中华学习机小蜜蜂—Ⅰ 使用与技术参考手册

清华大学出版社